The Number Field Sieve for Barreto-Naehrig Curves: Smoothness of Norms

by

Michael Shantz

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Combinatorics and Optimization

Waterloo, Ontario, Canada, 2015

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

The security of pairing-based cryptography can be reduced to the difficulty of the discrete logarithm problem (DLP) in finite fields of medium characteristic. The number field sieve is the best known algorithm for this problem. We look at a recent improvement to the number field sieve (NFS) by Joux and Pierrot that applies to finite field DLPs arising from elliptic curves used in pairing-based cryptography. We give specific parameter values for use with Miyaji-Nakabayashi-Takano curves offering 80-bits of security, and Barreto-Naehrig (BN) curves offering 128-bits of security. The running times of the corresponding NFS implementations are compared to the running times arising from prior versions of the NFS, showing that for BN curves the Joux-Pierrot version of the NFS is faster than the conventional version, but that BN curves still provide 128-bits of security. To get a better estimate on the number of relations that can be obtained during the sieving stage, we then analyze the distribution of the sizes of the product of the norms. Using this data, we give some guidelines for choosing which Joux-Pierrot polynomials to use for a specific DLP instance. We attempt to find a model for the distribution in order to further improve on the Joux-Pierrot version of the NFS. Finally, we prove some tighter bounds on the product of the norms.

Acknowledgements

I would like to thank my supervisor, Edlyn Teske, for her guidance and patience throughout my thesis process, and for getting me started in cryptography research during my undergrad. I would also like to thank Manuel Charlemagne for providing code, and for contributing to valuable discussions about his paper. Finally, I wish to thank my readers, Alfred Menezes and David Jao, for their insight and comments.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

In recent years, there have been several interesting cryptographic protocols developed based on bilinear pairings. These include a three-party one-round key exchange protocol [27], a short signature scheme [11] and an identity-based encryption scheme [10], all of which make use of bilinear pairings based on elliptic curves.

Evaluating the security of these protocols is an important cryptographic problem. Given an additive group G_1 and a multiplicative group G_2, a bilinear pairing is a special type of map $e : G_1 \times G_1 \rightarrow G_2$. The security of each of the three protocols above reduces to solving the discrete logarithm problem (DLP) in either G_1 or G_2. The Tate pairing is a bilinear pairing on (G_1, G_2), where G_1 is a subgroup of the group of points on an elliptic curve, and G_2 is a subgroup of the multiplicative group of a finite field. The security of pairing-based cryptography is thus based on the hardness of the DLP in both elliptic curve subgroups and in finite fields.

Although the best known algorithm for the elliptic curve DLP takes fully exponential time, the DLP in finite fields can be solved in less than fully exponential time using index-calculus based algorithms. We will focus on a finite field DLP algorithm known as the number field sieve (NFS). In particular, we will look at the version of the NFS that was shown by Joux *et al.* [28] to work for all finite fields.

The running time of the NFS for particular instances is notoriously difficult to accurately estimate. One approach is to examine an algebraic quantity known as the norm that is associated to all elements of number fields. The NFS requires finding a number of equations known as relations, and the size of an element's norms affects the probability that the element will provide a relation. By looking at the distribution of the sizes of the norms of elements, it is possible to find good parameters for particular NFS problems.

In this thesis, we will attempt to find good parameters and running time estimates for certain types of finite field DLPs. We will study two classes of elliptic curves which are recommended for use with pairing-based cryptography, namely Miyaji-Nakabayashi-Takano (MNT) [36] and Barreto-Naehrig (BN) [8] curves, and analyze how the NFS can be used to solve their related DLPs.

We also consider an improvement to the NFS made by Joux and Pierrot [30]. Their modification is only valid for certain types of finite fields, which fortunately include the finite fields that arise from the Tate pairing on BN curves.

In Chapter 2, we give background information needed for the rest of the thesis. We discuss some pairing-based cryptography protocols and their security. The Tate pairing for elliptic curves is defined, as are MNT and BN curves. Finally, we provide an overview of the NFS and describe the specific version that we will be working with.

Chapter 3 considers how the choice of various parameters can influence the running time of the NFS. We look at various recent versions of the NFS and discuss how to choose the best version for a particular DLP instance. We give some basic bounds on the norm that lead to running time estimates for the NFS. We also give specific parameter values for the types of DLPs arising from MNT and BN curves. Lastly, we look at how to choose parameters for the NFS modification of Joux and Pierrot, and give a running time analysis.

In Chapter 4 we collect data on the norms of elements. To better understand how many relations we will be able to gather, we look at how the sizes of these norms are distributed. We start by studying MNT curves, following the analysis done by Benger *et al.* [9]. We then extend their analysis to BN curves and look at how the choice of different NFS parameters, such as the characteristic and degree of the finite fields being used, can influence the distribution of the norm sizes. Based on this information, we propose a method that can be used to pick better parameters for specific DLPs.

Our goal in Chapter 5 is to find a good model to describe the distribution of norms. We again extend the work done by Benger *et al.* [9] and employ the Box-Cox method to find a good model. In particular, we find a polynomial T such that the fraction Y of elements with norms bounded above by X is given by $T(Y)$. The NFS requires working in two different fields and calculating norm values for both fields. We provide a model for the norm distribution in each field. We then discuss how our model could potentially be used to improve the running time of the NFS.

Finally, the focus of Chapter 6 is to find better theoretical bounds on the norm. The data collected in Chapter 4 suggests that our earlier norm bounds are not very tight. We prove some bounds on determinants that lead to tighter norm bounds.

We conclude in Chapter 7 with a summary of the most significant results from this thesis. Finally, in Chapter 8 we give some open problems arising from our work that could merit additional research.

Chapter 2

Preliminaries

In this section we provide the mathematical background necessary to understand the number field sieve and its connection to bilinear pairings. We give an introduction to pairing-based cryptography, followed by a short discussion of elliptic curves and the Tate pairing, before finally introducing the number field sieve.

First, we list some basic terminology which will be used through this paper. For a prime power q, let $\mathbb{F}_q$ denote the finite field of size q. Given two functions $f(x)$ and $g(x)$, we write $f(x) = O(g(x))$ if there exists $0 < M \in \mathbb{R}$ and $x_0 \in \mathbb{R}$ such that $|f(x)| \leq M\,|g(x)|$ for all $x > x_0$. Similarly, we write $f(x) = o(g(x))$ if for all $0 < M \in \mathbb{R}$ there exists a constant $x_M \in \mathbb{R}$ such that $|f(x)| \leq M\,|g(x)|$ for all $x > x_M$

An algorithm for which there is no known polytime algorithm, but which runs in better than fully exponential time is said to be a **subexponential** algorithm. To better specify the running time of a subexponential algorithm, the following L_q notation is often used. Given a prime power q and two constants $\alpha \in [0, 1]$ and c, let

$$L_q(\alpha, c) = \exp((c + o(1))(\log q)^{\alpha}(\log\log q)^{1-\alpha}).$$

When working with specific q and α values, we will sometimes abuse this notation and write

$$L_q(\alpha, c) = \exp(c(\log q)^{\alpha}(\log\log q)^{1-\alpha})$$

in order to find an exact value for c.

2.1 Pairing-Based Cryptography

In recent years, there have been multiple advances in the field of pairing-based cryptography. In this section, we explain what is meant by a bilinear pairing and give examples of popular pairing-based protocols and the number theoretic problems on which their security is based. In particular, we will discuss Joux's three-party one-round key agreement protocol, the BLS signature scheme and the identity-based encryption scheme of Boneh and Franklin.

Let r be a prime, let $G_1 = \langle P \rangle$ be an additive cyclic group of order r and let G_2 be a multiplicative group of order r. A **bilinear pairing** on (G_1, G_2) is an efficiently computable map $e : G_1 \times G_1 \to G_2$ such that

1. (bilinearity) $\forall\, R, S, T \in G_1, \quad e(R+S, T) = e(R, T) + e(S, T)$ and $e(R, S+T) = e(R, S) + e(R, T)$, and
2. (non-degeneracy) $e(P, P) \neq 1$.

The first pairing-based cryptographic protocol was the three-party one-round key agreement protocol developed by Joux in 2000 [27]. As suggested by its name, this protocol allows three parties to establish a shared secret key in a single round of communications. It is conceptually similar to the famous Diffie-Hellman key establishment protocol developed in 1976 [19]. The security of Diffie-Hellman is based on the following number theoretic problem, known as the **Diffie-Hellman problem** or DHP: given an additive group G and elements $P, aP, bP \in G$, calculate abP.

Joux's key establishment protocol provides security against passive adversaries based on the intractability of the following variation of the DHP. Given a bilinear pairing e on $(G_1 = \langle P \rangle, G_2)$, and four elements P, aP, bP and $cP \in G_1$, the **bilinear Diffie-Hellman problem**, or BDHP, is to compute $e(P, P)^{abc}$.

A second key application of bilinear pairings is the signature scheme of Boneh, Lynn and Shacham (BLS) proposed in 2001 [11]. The scheme makes use of a bilinear pairing on (G_1, G_2). Forging a signature in this scheme requires solving an instance of the **discrete logarithm problem** or DLP in G_1. The problem is as follows: given an element $R \in G_1 = \langle P \rangle$, find the smallest non-negative integer l, denoted $\log_P R$, such that $R = lP$.

Many other signature schemes based on the DLP have been proposed in the past, most notably the ElGamal signature scheme from 1985 [20]. The advantage of the BLS scheme is that it uses a single group element as its signature, whereas previous schemes normally required two group elements. This allows the BLS scheme to use signatures with approximately half the size of those used in other related signature schemes.

The final application of bilinear pairings that we will discuss is the identity-based encryption scheme of Boneh and Franklin from 2001 [10]. The concept of identity-based encryption was first proposed by Shamir in 1984 [48]. It simplifies the procedure for distributing authenticated public keys by allowing a user to use some form of simple identifying information, such as their email address, as their public key. The security of Boneh and Franklin's scheme is based on the difficulty of the BDHP.

We conclude this section by mentioning some reductions between the three problems defined above. First, it is clear that the DHP in a group G reduces to the DLP in G. Next, given a bilinear pairing on (G_1, G_2), note that the DLP in G_1 can be reduced to the DLP in G_2 since if $S = lP \in G_1$ then $e(S, P) = e(P, P)^l \in G_2$. Using bilinearity, it can also easily be shown that the BDHP for a bilinear pairing on (G_1, G_2) can be reduced to solving either the DLP in G_1 or the DLP in G_2.

For more details on bilinear pairings, see the survey paper by Menezes [33].

2.2 Elliptic Curves and the Tate Pairing

The bilinear pairings used in the protocols described above make use of elliptic curve groups. In this section, we first give a brief summary of key facts about elliptic curves, then discuss the best known algorithms for solving elliptic curve DLPs, before finally defining the Tate pairing and describing certain classes of curves with low embedding degree. Much of this section is based on the survey paper by Menezes [33].

A non-singular **elliptic curve** E over a field K of characteristic $p > 3$ is defined by a Weierstrass equation of the form

$$y^2 = x^3 + ax + b \tag{2.1}$$

where $a, b \in K$ such that $4a^3 + 27b^2 \neq 0$. The set $E(K)$ of points on the curve consists of all points $(x, y) \in K \times K$ that satisfy the Weierstrass equation, as well as an additional point at infinity, denoted ∞. For any finite field, we can define a group law on these points to convert them into a finite additive group with ∞ as the identity element. For $q = p^s$, Hasse's theorem can be used to bound the number of points on the curve by $\#E(\mathbb{F}_q) = q+1-t$, where $|t| \leq 2\sqrt{q}$. A curve E for which $p \mid t$ is said to be **supersingular**. A curve which is not supersingular is said to be **ordinary**.

Given an elliptic curve $E(K)$, a point $P \in E(K)$ of order N and a point $R \in \langle P \rangle$, the **elliptic curve discrete logarithm problem**, or ECDLP, is to find the lowest non-negative integer l such that $R = lP$. For general elliptic curves, there is no known subexponential time algorithm for the ECDLP. The best known algorithm is Pollard's rho algorithm [42], which has expected running time $O(\sqrt{N})$. This is fully exponential in $\log q$ when $N \approx q$.

For some elliptic curves, there is another ECDLP algorithm which can be faster than Pollard's rho. The key to this algorithm is to construct a bilinear pairing from $E(K)$ to some other group in which the DLP is easier. As noted in Section 2.1, the DLP in $E(K)$ then reduces to the DLP in this easier group.

Given a field K, let $\overline{K}$ denote the algebraic closure of K. Given an elliptic curve $E(\mathbb{F}_q)$, and a point P on the curve of prime order r with $\gcd(r, q) = 1$, let μ_r denote the group of r-th roots of unity in $\overline{\mathbb{F}_q}$. The **embedding degree of** $E(\mathbb{F}_q)$ **with respect to** r is the extension degree $[\mathbb{F}_q(\mu_r) : \mathbb{F}_q]$. It can be easily calculated, as it is also equal to the lowest positive integer k such that $r \mid q^k - 1$. For any elliptic curve E used in cryptography, we may assume that $\#E(K)$ has a single large prime factor r. As such, we define the **embedding degree** k of a curve E as the embedding degree of E with respect to r.

The Tate pairing allows us to reduce the ECDLP to the DLP in $\mathbb{F}_{q^k}^*$ [22]. For curves with a low embedding degree, certain index-calculus algorithms can solve the DLP in $\mathbb{F}_{q^k}^*$ in subexponential time. Note that for an arbitrary elliptic curve, we expect that $k \approx r$, in which case the corresponding DLP instance in $\mathbb{F}_{q^k}$ is too large to be practical.

All pairing-based protocols based on elliptic curves require working in $\mathbb{F}_{q^k}$. In order for these calculations to be done efficiently, we must choose curves with relatively low embedding degree, making them vulnerable to these index-calculus attacks. All supersingular elliptic curves have embedding degree $k \leq 6$ [34]. Two other important classes of curves with low embedding degree will be discussed in Section 2.2.1. Describing the number field sieve, a family of index-calculus type algorithms for the DLP in $\mathbb{F}_{q^k}^*$, will be the focus of Section 2.3.

We now define a few more concepts related to elliptic curves that will enable us to define the Tate pairing. A **divisor** D of an elliptic curve $E(\overline{K})$ is a formal sum of the form

$$D = \sum_{P \in E(\overline{K})} n_P(P),$$

where the n_P are integers with $n_P = 0$ for all but finitely many n_P. The **support** of a divisor D is the finite set of points $P \in E(K)$ for which $n_P \neq 0$. A divisor $D =$

$\sum_{P \in E(\overline{K})} n_P(P)$ is defined over K if $D^\sigma = \sum_{P \in E(\overline{K})} n_P(P^\sigma) = D$ for all K-automorphisms σ of $\overline{K}$. The set of all divisors defined over K is denoted by $\mathrm{Div}_K(E)$.

The **function field** $K(E)$ of E over K is the field of fractions of $K[x, y]/(f(x, y))$, where $f(x, y) = 0$ is the Weierstrass equation of E over K. Given an element $a \in K[x, y]/(f(x, y))$ and a point $P \in E$, let $\mathrm{ord}_P(a)$ denote the multiplicity of P as a root of a. We can extend ord_P to a well-defined valuation on $K(E)$ as follows: for $a, b \in K[x, y]/(f(x, y))$, let $\mathrm{ord}_P(\frac{a}{b}) = \mathrm{ord}_P\, a - \mathrm{ord}_P\, b$. We now define the **divisor** of $g \in K(E)$ to be the divisor

$$\mathrm{div}(g) = \sum_{P \in E} \mathrm{ord}_P(g)(P).$$

Note that $\mathrm{div}(g) \in \mathrm{Div}_K(E)$ for all $g \in K(E)$.

To simplify the presentation of the Tate pairing, we will assume that we are working with a point P of prime order r and embedding degree $k > 1$ such that $r^2 \nmid \#E(\mathbb{F}_q)$ and $r^3 \nmid \#E(\mathbb{F}_{q^k})$. Let $E[r]$ denote the set of all points $P \in E(\overline{\mathbb{F}_q})$ whose order divides r. Given a function $f \in K(E)$ and a a divisor $D = \sum n_P(P) \in \mathrm{Div}_K(E)$ such that $\mathrm{div}(f)$ and D have disjoint support, the field element $f(D) \in K$ is given by $f(P)^{n_P}$. We now define the **Tate pairing** to be a map

$$e : E[r] \times E[r] \to \mu_r$$

of the form

$$e(Q, R) = \left(\frac{f_Q(R + S)}{f_Q(S)} \right)^{(q^k - 1)/r},$$

where $Q, R, S \in E[r]$, f_Q is a function with $\mathrm{div}(f_Q) = r(Q) - r(\infty)$ and μ_r is the order r subgroup of $\mathbb{F}_{q^k}^*$. We also want $S \in E[r] \setminus \{\infty, Q, -R, Q - R\}$ so that $(R + S) - S$ and $\mathrm{div}(f_Q)$ have disjoint support. Note that the value of $e(Q, R)$ does not depend on the choice of S or f_Q. A function f_Q of the above form can always be found efficiently using Miller's algorithm [35]. This algorithm can also be used to compute the Tate pairing in polynomial time.

Although the Tate pairing is an efficiently computable, bilinear, non-degenerate map it is not a bilinear pairing since $E[r]$ is isomorphic to $\mathbb{Z}/r\mathbb{Z} \times \mathbb{Z}/r\mathbb{Z}$ and hence is not cyclic. For supersingular curves, this issue can be easily resolved through the use of a **distortion map** Ψ, which can be any endomorphism $\Psi : E \to E$ for which there exists a point $P \in E(\mathbb{F}_Q)$ such that $\Psi(P) \notin \langle P \rangle$. The map $\hat{e} : \langle P \rangle \times \langle P \rangle \to \mu_r$ defined by $\hat{e}(Q, R) = e(Q, \Psi(R))$ is then a bilinear pairing on $(\langle P \rangle, \mu_r)$.

For ordinary elliptic curves, no such distortion map exists [52]. However, the Tate pairing can still be modified to create a bilinear map suitable for use in pairing-based cryptography [33, Section 5.3].

2.2.1 Curves with Low Embedding Degree

In this section we discuss two classes of ordinary elliptic curves with low embedding degree that are suitable for use in pairing-based cryptography.

Techniques for constructing ordinary elliptic curves with low embedding degree make use of the complex multiplication (CM) method [2] [37]. Given an integer t and a prime p such that

$$t^2 - 4p = -DV^2, \tag{2.2}$$

where D is positive and squarefree if t is odd, and $D/4$ is positive and squarefree if t is even, the CM method can be used to find an elliptic curve E over $\mathbb{F}_p$ with prime order $N = \#E(\mathbb{F}_p) = p + 1 - t$. Although the CM method is not a polytime algorithm, it is still reasonably fast when D is small.

The first ordinary curves we will discuss are those discovered by Miyaji, Nakabayashi and Takano (MNT) in 2001 [36]. By letting $t = 1 \pm 2l$, $p = 4l^2 + 1$ and $U = 6l \pm 1$, solving Equation (2.2) reduces to solving the Diophantine equation $U^2 - 3DV^2 = -8$. MNT curves with embedding degree 6 can be efficiently found by choosing a small value for D and finding a solution to the Diophantine equation with $U \equiv \pm 1 \pmod 6$ and N, p prime, then using the CM method.

A similar approach was taken in 2005 by Barreto and Naehrig (BN) [8]. To find an elliptic curve with embedding degree $k = 12$, let $t = 6z^2 + 1$ and $p = 36z^4 + 36z^3 + 24z^2 + 6z + 1$. The CM equation then becomes $t^2 - 4p = -3(1 + 4z + 6z^2)^2 = 3V^2$. The desired elliptic curve can be found by choosing random integers z of the appropriate size until both p and $N = p + 1 - t$ are prime, then using the CM method.

When implementing pairing-based protocols, there is a trade-off that must be considered when choosing the size of the embedding degree k. These protocols require doing calculations in $\mathbb{F}_{p^k}$, so it is preferable to have low embedding degree. However, recall that an ECDLP instance can be reduced to an instance of the DLP in $\mathbb{F}_{p^k}^*$. A lower embedding degree thus makes the protocol more vulnerable to index-calculus based methods.

Since the ECDLP can also simply be solved by running Pollard's rho method in $E(\mathbb{F}_p)$, we get three important security parameters to consider when evaluating the security of pairing-based protocols: the field size p, the embedding degree k and the elliptic curve group order N. We wish to balance the difficulty of the ECDLP and the DLP, keeping in mind that the best algorithm for the ECDLP takes time $O(\sqrt{N})$, while index-calculus methods solve the DLP in finite fields in subexponential time.

For the ordinary curves with low embedding degree described above, the embedding degree is fixed. We also have $N \approx p$. To achieve a given security level, say 80-bits, we need $N \geq 2^{160}$ so that $\sqrt{N} \geq 2^{80}$. We also need the DLP in $\mathbb{F}_{p^k}^*$ to offer 80-bits of security. The exact size of p needed to achieve this security is harder to estimate, but is based on the best known algorithm for the DLP. In this case, a value of $p^k \geq 2^{1024}$ is often used, since 2^{1024} is the field size listed by NIST as providing 80-bits of security for the finite field DLP [7, Table 2].

MNT curves offering 80-bits of security are often used since the difficulty of the ECDLP and DLP are well-balanced for $N \approx p \approx 2^{160}$ and $k = 6$. BN curves work well at the 128-bit security level, since picking $N \approx 2^{2 \cdot 128} = 2^{256}$ gives $p^k \approx 2^{256 \cdot 12} = 2^{3072}$, the field size that provides 128-bits of security according to NIST [7, Table 2].

Finally, note that the running time of index-calculus based methods for the DLP depend on the relative sizes of $\mathbb{F}_{p^k}$ and its characteristic p. The running times are better understood for the cases where either p or k is very small. Unfortunately, the finite fields $\mathbb{F}_{p^k}$ obtained from bilinear pairings fall in between these two cases, into the so-called **medium characteristic case**. A more detailed discussion on how the relative size of p^k and p affects the choice of discrete logarithm algorithms is provided in Section 3.1.

2.3 The Number Field Sieve

The **number field sieve**, or NFS, is a class of algorithms, first developed in the early 1990's to solve the integer factorization problem [31] [49]. It was shown by Gordon in 1993 [23] that the ideas in the NFS could be used to obtain an index-calculus based algorithm to solve the DLP in finite fields. In particular, given a prime power $q = p^n$, a primitive element $s \in \mathbb{F}_q^*$ and an element u in $\mathbb{F}_q^*$, the NFS can be used to find $\log_s u$. In this section we provide the mathematical background from algebraic number theory needed to understand the NFS. We then give an overview of the algorithm and provide details on the version of the algorithm that we will be using.

2.3.1 Algebraic Number Theory

Before introducing the key ideas behind the NFS, we first require a few concepts from algebraic number theory [50]. A **number field** is a finite field extension of $\mathbb{Q}$. An **algebraic integer** is a root of a monic polynomial in $\mathbb{Z}[x]$. Let $\mathbb{A}$ denote the set of all algebraic integers. The **ring of integers** of a number field K is the subring $\mathcal{O}_K = K \cap \mathbb{A}$ of K whose elements are the algebraic integers in K.

Not every ring of integers is a unique factorization domain. However, it is a basic fact in algebraic number theory that every ring of integers $\mathcal{O}$ is a Dedekind domain, which implies that every nonzero ideal in $\mathcal{O}$ has a unique factorization as a product of prime ideals in $\mathcal{O}$ [50, pp. 115-117].

Given an ideal I in some ring of integers $\mathcal{O}$, the **norm** of I is defined as

$$N(I) = |\mathcal{O}/I|,$$

the cardinality of the quotient ring $\mathcal{O}/I$.

Given a $1 \times k$ vector $\vec{a} = [a_1 \dots a_k]$, let $s(\vec{a}) = [a_k a_1 \dots a_{k-1}]$ denote the **right shift** of $\vec{a}$.

Let $f(x) = a_n x^n + \cdots + a_1 x + a_0 \in \mathbb{F}_p[x]$ and $g(x) = b_m x^m + \cdots + b_1 x + b_0 \in \mathbb{F}_p[x]$ be two polynomials of degrees n and m respectively. Let $\vec{f}$ be the $1 \times (m+n)$ vector whose i^{th} coordinate is a_{n+1-i} for $i = 1, \dots, n+1$, and whose remaining $m-1$ coordinates are zero. Likewise let $\vec{g}$ be the $1 \times (m+n)$ vector whose i^{th} coordinate is b_{m+1-i} for $i = 1, \dots, m+1$, and whose remaining $n-1$ coordinates are zero.

The **Sylvester matrix** of f and g is the $(n+m) \times (n+m)$ matrix given by

$$\begin{bmatrix} \vec{f} \\ s(\vec{f}) \\ \vdots \\ s^{m-1}(\vec{f}) \\ \vec{g} \\ s(\vec{g}) \\ \vdots \\ s^{n-1}(\vec{g}) \end{bmatrix}. \tag{2.3}$$

For example, if $f(x) = a_3 x^3 + a_2 x^2 + a_1 x + a_0$ and $g(x) = b_2 x^2 + b_1 x + b_0$, then the Sylvester matrix of f and g would be

$$\begin{bmatrix} a_3 & a_2 & a_1 & a_0 & 0 \\ 0 & a_3 & a_2 & a_1 & a_0 \\ b_2 & b_1 & b_0 & 0 & 0 \\ 0 & b_2 & b_1 & b_0 & 0 \\ 0 & 0 & b_2 & b_1 & b_0 \end{bmatrix}.$$

The **resultant** of f and g, denoted $\mathrm{Res}_x(f,g)$, is defined as the determinant of the Sylvester matrix of f and g.

Given a number field $K = \mathbb{Q}[x]/(f(x))$ of degree n over $\mathbb{Q}$, and an element $\alpha = \sum_{j=0}^{n-1} a_j x^j + (f(x)) \in K$, the **norm over** $\mathbb{Q}$ of α is given by

$$N_K(\alpha) = \mathrm{Res}_x\left(\sum_{j=0}^{n-1} a_j x^j, f(x)\right).$$

Note that the norm is a multiplicative function and that if α is an algebraic integer, then $|N_K(\alpha)| = N((\alpha))$ [50, pp. 126-127]. Further, every prime ideal has norm of the form ρ^k for some prime ρ and integer $k \leq n$ [50, p. 129]. A prime ideal P with norm ρ^k is said to **lie above** ρ. As a consequence of these two facts, we get that an ideal I has a prime factor P lying over a prime ρ if and only if $N(I)$ is divisible by ρ.

Finally, we note that the prime factorization of an ideal can be easily determined once the prime factorization of its norm is known [28, Sec. 4.1] [16, Sec. 4.8].

2.3.2 Overview of the Number Field Sieve

We now provide an overview of how the NFS can be used to solve the DLP, based on the presentation of Schirokauer [47].

In the NFS, we will work with two subrings of rings of integers, $R_1 \subseteq \mathcal{O}_1$ and $R_2 \subseteq \mathcal{O}_2$, as well as a pair of homomorphisms $\psi_i : R_i \to \mathbb{F}_q$ for $i = 1, 2$, where p is a prime and $q = p^n$. The goal of the NFS is to find an integer y and a pair of $(q-1)$-th powers $\beta_1 \in R_1$ and $\beta_2 \in R_2$ such that

$$\psi_1(\beta_1) = \psi_2(\beta_2) \cdot s^y u. \tag{2.4}$$

It then follows that $s^y u = \frac{\psi_1(\beta_1)}{\psi_2(\beta_2)}$ is also a $(q-1)$-st power in $\mathbb{F}_q$. Thus $y \equiv -\log_s u \pmod{q-1}$, providing a solution to the DLP.

Note that in practice we will work with a large odd factor l of $q-1$, and find two l-th powers of the above form. The reasons for this are technical and will be discussed briefly near the end of this section. For now, suffice it to note that although the following algorithm can be modified to work modulo any prime power l [45], there are other algorithms that are faster in practice when working with a power of a small prime [43].

To find the pair β_1, β_2, we want to be able to find lots of pairs of elements $\alpha_1 \in R_1$ and $\alpha_2 \in R_2$ such that $\psi_1(\alpha_1) = \psi_2(\alpha_2)$. To do so, we let $f_1, f_2 \in \mathbb{Z}[x]$ be a pair of irreducible polynomials of degree $n_1, n_2 \geq n$ respectively, having small coefficients and with a common irreducible factor g of degree n over $\mathbb{F}_p$. Let δ be a root of g in $\mathbb{F}_q$. For $i = 1, 2$, let θ_i be a root of f_i and let $K_i = \mathbb{Q}[x]/(f_i(x)) = \mathbb{Q}(\theta_i)$ be a number field of degree n_i.

Let $R_i = \mathbb{Z}[\theta_i]$ be a subring of the ring of integers $\mathcal{O}_i$ of K_i, and let $\psi_i : R_i \to \mathbb{F}_q$ be the homomorphism given by

$$\psi_i\left(\sum_{j=0}^{n_i-1} a_j \theta_i^j\right) = \sum_{j=0}^{n_i-1} a_j \delta^j \pmod{p},$$

a diagram of which is given in Figure 2.1. Then for any polynomial $f \in \mathbb{Z}[x]$ of degree at most $\min(n_1, n_2)$, we have $\psi_1(f(\theta_1)) = \psi_2(f(\theta_2))$.

$$\begin{tikzcd}
R_i = \mathbb{Z}[\theta_i] \arrow[d, "a \mapsto a \pmod{p}"'] \arrow[dr, "\psi_i"] & \\
\mathbb{F}_p(\theta_i) \arrow[r, "\theta_i \mapsto \delta"'] & \mathbb{F}_q = \mathbb{F}_p(\delta)
\end{tikzcd}$$

Figure 2.1: Homomorphism $\psi_i : R_i \to \mathbb{F}_q$, where θ_i is a root of f_i and δ is a root of g.

The key to finding β_1 and β_2 is to find lots of pairs α_1, α_2 satisfying $\psi_1(\alpha_1) = \psi_2(\alpha_2)$, with the additional restriction that the principal ideals generated by α_1 and α_2 have all their prime factors contained in some fixed set $\mathcal{B}$, called the **factor basis**. An ideal satisfying this condition is said to be **$\mathcal{B}$-smooth**. Note that we must work with factorizations of ideals instead of factorizations of elements since we may not have unique factorization in the latter case. However, as mentioned in Section 2.3.1, every ideal in a ring of integers has a unique factorization as a product of prime ideals, so the notion of $\mathcal{B}$-smooth ideals is indeed well-defined. Given an element α in $\mathcal{O}_1$ or $\mathcal{O}_2$, we will also say that α is $\mathcal{B}$-smooth if the principal ideal (α) is $\mathcal{B}$-smooth.

In Section 2.3.3, we will give details on how to select a good factor basis. We will also discuss how to check if an ideal I is $\mathcal{B}$-smooth and how to find the factorization of $\mathcal{B}$-smooth ideals. For now, assume that these two things can be done in a reasonable amount of time.

In this section we will let $\mathcal{B} = \{P_1, \ldots, P_k, Q_1, \ldots, Q_m\}$ be a generic factor basis, where $P_1, \ldots, P_k$ are prime ideals in $\mathcal{O}_1$ and $Q_1, \ldots, Q_m$ are prime ideals in $\mathcal{O}_2$. Given a pair of $\mathcal{B}$-smooth elements (α_1, α_2), let $e_j(\alpha_1)$ denote the degree of P_j in the factorization of (α_1), for $j = 1, \ldots, k$, so that

$$(\alpha_1) = \prod_{j=1}^{k} P_j^{e_j(\alpha_1)}. \tag{2.5}$$

Similarly, for $j = 1, \ldots, m$, let $\nu_j(\alpha_j)$ denote the degree of Q_j in the factorization of (α_2), so that

$$(\alpha_2) = \prod_{j=1}^{m} Q_j^{\nu_j(\alpha_2)}.$$

The pair (α_1, α_2), together with the $1 \times (k+m)$ vector $\big(e_1(\alpha_1) \cdots e_k(\alpha_1)\, \nu_1(\alpha_2) \cdots \nu_m(\alpha_2)\big)$, is called a **relation**.

The first part of the NFS, commonly referred to as the **sieving stage** (or relation gathering stage), simply involves collecting slightly more than $|\mathcal{B}|$ relations. The second part is the **linear algebra stage**, in which we attempt to combine our relations in order to construct a pair β_1, β_2 as defined in Equation (2.4).

To simplify the explanation of the linear algebra stage, we will assume for now that we are given $\mathcal{B}$-smooth elements $\tau, \upsilon \in R_1$ such that $\psi_1(\tau) = s$ and $\psi_1(\upsilon) = u$. We then define V_τ to be the $1 \times (k+m)$ vector $\big(e_1(\tau) \cdots e_k(\tau)\, 0 \cdots 0\big)$, where the e_j are defined as in Equation (2.5). Also define V_υ correspondingly.

Now consider the system

$$A\vec{x} \equiv -V_\upsilon \pmod{l}, \tag{2.6}$$

where A is the matrix whose first column is the vector V_τ and whose remaining columns are the vectors associated with each relation generated in the sieving stage. Having generated slightly more than $|\mathcal{B}|$ relations, we should be able to find a solution $\vec{x}$ with high probability. If we are unsuccessful, we merely generate a few more relations and try again.

Finding solutions to Equation 2.6 remains a non-trivial problem. We would like to complete the linear algebra stage in roughly the same amount of time as the sieving stage. It turns out that basic Gaussian elimination is too slow, so special methods are required. Fortunately, the system is very sparse and can be turned into a much smaller dense matrix. The Wiedemann algorithm [53] has a sufficiently fast expected running time, while structured Gaussian elimination, the Lanczos algorithm, and the conjugate gradient algorithm [39] [17] are all effective in practice. Although the linear algebra stage of the integer factorization version of the NFS has been parallelized, this remains a difficult problem for the DLP version over fields with characteristic $p > 2$. In running time analyses of the

NFS, the running time of the linear algebra stage is generally taken to be $|\mathcal{B}|^2$. For a more thorough discussion of methods for the linear algebra stage, refer to [38].

Given a solution $\vec{x}$ to the above system, let x_τ denote the first entry of $\vec{x}$. For each relation (α_1, α_2), let x_{α_1,α_2} denote the entry of $\vec{x}$ associated to (α_1, α_2). Let

$$\gamma = \tau^{x_\tau} \upsilon \prod_{\text{relations } (\alpha_1,\alpha_2)} \alpha_1^{x_{\alpha_1,\alpha_2}} \in \mathcal{O}_1$$

and

$$\epsilon = \prod_{\text{relations } (\alpha_1,\alpha_2)} \alpha_2^{x_{\alpha_1,\alpha_2}} \in \mathcal{O}_2.$$

Then $e_j(\gamma) \equiv 0 \pmod{l}$ for $j = 1, \ldots, k$ and $\nu_\iota(\epsilon) \equiv 0 \pmod{l}$ for $\iota = 1, \ldots, m$. Hence the ideals (γ) and (ϵ) are both l-th powers of some ideals $I_1 \subseteq \mathcal{O}_1$ and $I_2 \subseteq \mathcal{O}_2$ respectively.

Recall that our goal is to find a pair of l-th powers $\beta_1 \in R_1$ and $\beta_2 \in R_2$ satisfying Equation (2.4). Also, each relation (α_1, α_2) satisfies $\psi_1(\alpha_1) = \psi_2(\alpha_2)$ by construction. Thus

$$\begin{aligned}\psi_1(\gamma) &= \psi_1(\tau^{x_\tau})\psi_1(\epsilon) \prod \psi_1(\alpha_1^{x_{\alpha_1,\alpha_2}}) \\ &= s^{x_\tau} u \prod \psi_2(\alpha_2^{x_{\alpha_1,\alpha_2}}) \\ &= s^{x_\tau} u \, \psi_2(\epsilon).\end{aligned}$$

Hence if γ and ϵ were l-th powers of elements in R_1 we would be done.

For large l, we expect that both I_1 and I_2 will be principal ideals, generated by $\eta_1 \in \mathcal{O}_1$ and $\eta_2 \in \mathcal{O}_2$ respectively [47, p. 402]. It then follows that $\gamma = \eta_1^l \omega_1$ and $\epsilon = \eta_2^l \omega_2$ for some units $\omega_1 \in \mathcal{O}_1$ and $\omega_2 \in \mathcal{O}_2$. The case where either I_1 or I_2 are not principal is more complicated and will not be discussed here, although it does not present any major impediment [28, Sec 4.2].

Unfortunately, the algorithm as described above does not guarantee that ω_1 and ω_2 are l-th powers. The solution is to add some more information to each of our relations. For each relation obtained in the sieving stage, we calculate the value of several **Schirokauer maps** and add these values to the end of the relation's exponent vector. In the linear algebra stage, we then solve this slightly larger systems of equations. With high probability, the resulting solution generates γ and ϵ values that are indeed l-th powers in $\mathcal{O}_1$ and $\mathcal{O}_2$. For more information on Schirokauer maps, refer to the paper by Schirokauer [47, pp. 401-402].

There are however two more important facts about Schirokauer maps that bear mentioning. First, the number of Schirokauer maps for K_i needed is much less than the typical size of $\mathcal{B}$, so the use of Schirokauer maps does not significantly influence the running time of either the sieving stage or the linear algebra stage of the NFS. Second, in order to efficiently calculate the value of these Schirokauer maps, we need to be working modulo a prime. For this reason, it is necessary to work with a large prime factor l of $q-1$, rather than with $q-1$ itself.

Finally, although we are able to construct l-th powers γ and ϵ of elements in $\mathcal{O}_1$ and $\mathcal{O}_2$ respectively, the l-roots of γ and ϵ may not be elements of R_1 and R_2 respectively. Fortunately, this problem has an easy fix. Let f_i' denote the formal derivative of the polynomial f_i for $i = 1, 2$. Since $f_i'(\theta_i)\mathcal{O}_i \subseteq R_i$ for $i = 1, 2$, it follows that $f_1'(\theta_1)^l\gamma$ and $f_2'(\theta_2)^l\epsilon$ are l-th powers of elements in R_1 and R_2 respectively, as required.

2.3.3 The Sieving Stage and Choice of a Factor Basis

In order to come up with implementations of the NFS with a good running time, both experimentally and theoretically, we need to do two things. First, we need an effective method for finding relations. It is also important to choose a good factor basis, since the size of the factor basis directly influences the running time of both the sieving and linear algebra stages.

To generate relations, we pick subsets $\mathcal{S}_1 \subseteq R_1 = \mathbb{Z}[\theta_1]$ and $\mathcal{S}_2 \subseteq R_2 = \mathbb{Z}[\theta_2]$ from which we expect to be able to find lots of relations. Following the implementation described in Joux *et al.* [28], we will work with a set of polynomials

$$\mathcal{R} = \left\{ \sum_{j=0}^{t} a_j x^j \in \mathbb{Z}[x] : 0 \leq a_j < S \text{ for } j = 0, \ldots, t \right\},$$

called the **sieving space**, where S is a constant called the **sieving bound**, and t is a constant less than $\min(n_1, n_2)$ that we will call the **sieving degree**. For $i = 1, 2$, our subset $\mathcal{S}_i$ of R_i is then given by $\varphi_i(\mathcal{R}) \subseteq \mathbb{Z}[\theta_i] \subseteq \mathcal{O}_i$, where $\varphi_i : \mathbb{Z}[x] \to \mathbb{Z}[\theta_i]$ is simply the evaluation homomorphism given by $\varphi_i(f(x)) = f(\theta_i)$. The degree constraint in our choice of a sieving space guarantees that φ_i is injective and hence that $|\mathcal{R}| = S^{t+1}$.

To generate relations, we repeatedly pick $\alpha \in \mathcal{R}$, uniformly at random. We then check to see if $(\varphi_1(\alpha), \varphi_2(\alpha))$ generates a relation. Clearly $\psi_1(\varphi_1(\alpha)) = \psi_2(\varphi_2(\alpha))$, as illustrated in Figure 2.2, so it suffices to check that $\varphi_i(\alpha)$ is $\mathcal{B}$-smooth for $i = 1, 2$.

Choosing a sieving space comprised of polynomials with coefficients less than S allows us to pick a relatively small factor basis. Let our factor basis $\mathcal{B}$ be the set of all prime

$$\begin{tikzcd}
& \mathbb{Z}[x] \arrow[dl, "\varphi_1"'] \arrow[dr, "\varphi_2"] & \\
R_1 = \mathbb{Z}[\theta_1] \arrow[dr, "\psi_1"'] & & R_2 = \mathbb{Z}[\theta_2] \arrow[dl, "\psi_2"] \\
& \mathbb{F}_q = \mathbb{F}_p(\delta) &
\end{tikzcd}$$

Figure 2.2: Equivalence of $\psi_1 \circ \varphi_1$ and $\psi_2 \circ \varphi_2$.

ideals P in $\mathcal{O}_1$ or $\mathcal{O}_2$ lying above a prime less than B, where B is a constant called the **smoothness bound**. We say that an ideal I in $\mathcal{O}$ is B**-smooth** if all its prime ideal factors lie above a prime less than B. Thus an ideal in $\mathcal{O}_1$ or $\mathcal{O}_2$ is $\mathcal{B}$-smooth if and only if it is B-smooth. We also say that an integer is B**-smooth** if all of its prime factors are less than B. It then follows from Section 2.3.1 that an ideal P is $\mathcal{B}$-smooth if and only if its norm is B-smooth.

Since we will only ever work with norms of elements in our sieving space, we abuse notation and refer to the norms of $\varphi_1(\alpha)$ and $\varphi_2(\alpha)$ as the norms of α. Additionally, we write $N_{K_i}(\alpha)$ instead of $N_{K_i}(\varphi_i(\alpha))$, for $i = 1, 2$.

In summary, to check whether a pair $(\varphi_1(\alpha), \varphi_2(\alpha))$ is $\mathcal{B}$-smooth it suffices to check that both $N_{K_1}(\alpha)$ and $N_{K_2}(\alpha)$ are B-smooth. Note that in practice we will use the factorization of $N_{K_i}(\alpha)$ to find the factorization of $(\varphi(\alpha))$, instead of attempting to find the factorization of the ideal directly.

To find an estimate for the running time of the NFS, we will need to know an upper bound on the size of our chosen factor basis. The number of ideals in our factor basis can be bounded by tB [46, p. 1276], giving us our desired upper bound. Since the linear algebra stage takes times $|\mathcal{B}|^2$, our choice of a factor basis gives a running time of t^2B^2, This is often simplified to B^2, since it is assumed that the size of t is insignificant compared to the size of B.

There is one final problem with the algorithm as presented above. Recall that in Section 2.3 we assumed that the elements s and u had B-smooth preimages in $\mathcal{O}_1$. Unfortunately, given a random instance of the discrete logarithm problem in $\mathbb{F}_q$ there is no reason why these smooth preimages need exist. To find discrete logarithms of arbitrary elements, we follow the descent approach of Joux *et al.* [28]. First, pick integers a, b such that

$y = u^a s^b$ has a preimage $\psi_1^{-1}(y) \in \varphi_1(R) \subseteq \mathcal{O}_1$ whose norm is B_1-smooth, for some constant $B_1 > B$. Finding such an element y can be done in time negligible to the rest of the NFS. Next, use the factorization of $N(\psi_1^{-1}(y))$ to factor $\psi_1^{-1}(y)$. For any factors which are not B-smooth, repeat the above procedure using a new bound B_2 such that $B_1 > B_2 \geq B$. By repeating this procedure and properly choosing the bounds $B_1, B_2, \dots B_k = B$, it is possible to find the discrete logarithm of any element without changing the asymptotic running time of the NFS.

2.3.4 Summary of the Number Field Sieve

Before we proceed to choose parameters to optimize the running time of the NFS, we give a brief summary of the algorithm. Recall that the goal of the NFS is to find $\log_s u$, where $s, u \in \mathbb{F}_q^*$ and s is a primitive element. To start, we pick two number fields K_1 and K_2, a smoothness bound B and a sieving space $\mathcal{R}$ comprised of all polynomials in $\mathbb{Z}[x]$ of degree at most the sieving degree t whose coefficients are all non-negative integers less than the sieving bound S. During the sieving stage of the NFS, we repeatedly select a random element of $\mathcal{R}$, calculate its K_1 and K_2 norms, and check if both norms are B-smooth. If so, we have found a relation. We continue doing this until we have collected as many relations as there are elements in our factor basis, which is at most tB. In the linear algebra stage, we then attempt to write u as a product of powers of s and the smooth sieving space elements found previously. From this equation, we can with high probability recover the discrete logarithm $\log_s u$.

Chapter 3

Parameter Choices

In this section we will determine the optimal values of various parameters used to implement the NFS for the finite field DLP related to bilinear pairings. We start by reviewing the method for choosing parameters provided by Joux *et al.* in their Appendix A.2 [28]. We then follow the approach of Benger *et al.* [9] and apply this method to choose parameters to use for the two classes of ordinary curves with low embedding degree discussed in Section 2.2.1, namely MNT curves and BN curves.

To optimize the running time of the NFS, we first need to determine approximate running times for the sieving stage and the linear algebra stage. We will then try to choose suitable parameters to roughly balance the running times of these two steps. In particular, we will find values for the sieving bound S, the smoothness bound B and the sieving degree t of the sieving space $\mathcal{R}$. Although specific implementations of the NFS will normally involve choosing f_1 and f_2 of some special form, we will at first derive some results that are valid for any appropriate choice of f_1 and f_2.

In the sieving stage, we repeatedly choose an element in our sieving space, then compute its norm and test it for smoothness. We do this until we have at least as many relations as the size tB of our factor basis. As such, we need B, S and t to be large enough for there to exist sufficiently many relations. To estimate the number of relations that we can find in the sieving step, we will assume that the probability of $N_{K_i}(\alpha)$ being B-smooth for a randomly chosen element $\alpha \in \mathcal{R}$ is approximately equal to the probability that a random integer of the same size is B-smooth, for $i = 1, 2$.

Next we wish to find a pair of upper bounds on the absolute value of the norm $|N_{K_i}(\alpha)|$ which is valid for all $\alpha \in \mathcal{R}$, for $i = 1, 2$. This will allow us to make use of the following theorem of Canfield, Erdős and Pomerance to estimate the numbers of elements in our sieving space with B-smooth norms [14]. Since every element of the sieving space with smooth norms corresponds to a relation, we thus get an estimate on the number of relations that we can expect to find in the sieving stage.

Theorem 3.1 ([14]). *A randomly chosen positive integer $m < L_q(r, c)$ will be $L_q(s, d)$-smooth with probability $L_q(r - s, -(r - s)\frac{c}{d})$.*

Recalling that the norm of α is simply the determinant of the Sylvester matrix given in Equation (2.3), we can derive a simple upper bound for the absolute value of the norm. Given a square matrix $M = [m_{ij}]_{1\leq i,j\leq m}$ let $M_{i,j}$ denote the submatrix of M obtained by removing the i-th row and j-th column.

Lemma 3.2. *Let A be an $m \times m$ matrix such that the absolute values of the entries in the i-th row of A are bounded above by B_i, for $i = 1, \ldots, m$. Then $|\det A| \leq m! \prod_{i=1}^{m} B_i$.*

Proof. If A is the 1×1 matrix $[a]$ then we can take $B_1 = |a|$ to get $|\det A| = |a| \leq 1!B_1$. Suppose the theorem holds for all $(m - 1) \times (m - 1)$ matrices. Let $A = [a_{ij}]_{1\leq i,j\leq m}$ be an $m \times m$ matrix such that the absolute values of the entries in the i-th row of A are bounded above by B_i, for $i = 1, \ldots, m$. From the Laplace expansion for the determinant, we get that

$$\begin{aligned}
|\det A| &= \left| \sum_{j=1}^{m} a_{1j}(-1)^{1+j} \det A_{1,j} \right| \\
&\leq \sum_{j=1}^{m} |a_{1j} \det A_{1,j}| \\
&\leq \sum_{j=1}^{m} \left(B_1 (m-1)! \prod_{i=2}^{m} B_i \right) \\
&= m! \prod_{i=1}^{m} B_i,
\end{aligned}$$

where the second inequality follows from induction since B_{i+1} is an upper bound on the absolute values of the entries in the i-th row of $A_{1,j}$, for $i = 1, \ldots, m-1$ and $j = 1, \ldots, m$. □

We can apply this theorem to Sylvester matrices to get an upper bound on the norm. For an element $\alpha = \sum_{j=0}^{t} a_j x^j + (f(x)) \in \mathcal{R}$, the norm $N_{K_i}(\alpha)$ is equal to the determinant of the Sylvester matrix A_i of f_i and $\sum_{j=0}^{t} a_j x^j$, for $i = 1, 2$. Let D_i be an upper bound on the absolute values of the coefficients of f_i, and let $n_i = \deg f_i$. Then the first t rows of A_i are bounded above by D_i, and the last n_i rows are bounded above by S. So we get an upper bound on the absolute value of the norm given by

$$|N_{K_i}(\alpha)| \leq (n + t)!\, S^{n_i} D_i^t. \tag{3.1}$$

For the product of the norms of α, we then get the following upper bound:

$$\begin{aligned} |N_{K_1}(\alpha) N_{K_2}(\alpha)| &\leq \left((n_1 + t)!\, S^{n_1} D_1^t\right) \left((n_2 + t)! S^{n_2} D_2^t\right) \\ &= (n_1 + t)!\, (n_2 + t)!\, S^{n_1 + n_2} (D_1 D_2)^t. \end{aligned} \tag{3.2}$$

Our goal is thus to chose B, S and t to make sure that we can generate enough relations, and to balance the running times of the sieving step and the linear algebra step. This will be the focus of Sections 3.4 to 3.7. Before doing so, we first give a brief overview of recent developments related to the NFS and mention some of the particular choices of polynomials f_1 and f_2 suggested, in particular those suggested for DLP instances arising from bilinear pairings.

3.1 Characteristic Size and Algorithm Choice

Determining the running time of an optimal NFS implementation is a difficult task. Let $q = p^n$ and consider the DLP in $\mathbb{F}_q$. The difficult of NFS running time estimates is in part due to the fact that the best version of the NFS to use for this problem is dependent on the relative sizes of p and q, as mentioned in Section 2.2.1. In this section we will give an overview of recent advances and address some of the issues regarding how to select the best algorithm for a specific DLP instance.

For very small p values (less than $L_q(\frac{1}{3}, c)$ for some small constant c), the quasi-polynomial algorithm of Barbulescu *et al.* [5] is the best choice. Larger p values fall into either the medium or high characteristic case. Although there is no precise boundary between these two cases, the high characteristic case generally refers to p values greater than $L_p(\frac{2}{3}, c)$ for c somewhere between 2 and 3. The medium prime case refers to p values between $L_q(\frac{1}{3}, c)$ and $L_q(\frac{2}{3}, c)$.

There has been a flurry of activity in the past couple years involving new algorithms for the medium and high characteristic cases. The running times of many of these algorithms are summarized in Table (3.1). These algorithms are based on the techniques introduced by Joux *et al.* in 2006 [28] that modified the NFS to run in time $L_q(\frac{1}{3}, c)$ in these two cases. They achieved an asymptotic running time of $L_q(\frac{1}{3}, \sqrt[3]{\frac{64}{9}}) \approx L_q(\frac{1}{3}, 1.923)$ for the high case and an asymptotic running time of $L_q(\frac{1}{3}, \sqrt[3]{\frac{128}{9}}) \approx L_q(\frac{1}{3}, 2.423)$ for the medium characteristic case. For the boundary of $p = L_q(\frac{2}{3}, c)$ between the medium and high cases, their algorithm had an asymptotic running time of $L_q(\frac{1}{3}, 2c')$ where $c' = \frac{4}{3}\left(\frac{3t}{4(t+1)}\right)^{\frac{1}{3}}$ and t is the positive integer closest to the real root of $3c^3t(t+1)^2 - 32 = 0$.

In 2014, Barbulescu and Pierrot improved on the work of Joux *et al.* by introducing the multiple number field sieve, or MNFS [6]. By working with more than two number fields, they improved the asymptotic running times of the medium and high characteristic cases to $L_q(\frac{1}{3}, (2^{13}/3^6)^{1/3}) \approx L_q(\frac{1}{3}, 2.240)$ and $L_q(\frac{1}{3}, ((92 + 26\sqrt{13})/27)^{1/3}) \approx L_q(\frac{1}{3}, 1.902)$ respectively.

Later in 2014, Barbulescu *et al.* made further improvements for the medium characteristic case [4]. Their conjugation method achieved an asymptotic running time of $L_q(\frac{1}{3}, \sqrt[3]{\frac{96}{9}}) \approx L_q(\frac{1}{3}, 2.201)$. By combining the MNFS with the conjugation method, Pierrot [41] was able to further improve this time to $L_q(\frac{1}{3}, (8(9 + 4\sqrt{6})/15)^{1/3})$.

In 2013, Joux and Pierrot provided a version of the number field sieve that offers improvements for the medium and high characteristic cases [30]. It is only applicable for finite fields with characteristic p that can be written as $p = P(u)$ for some polynomial $P \in \mathbb{Z}[x]$ of low degree λ with small coefficients. Fortunately, the fields resulting from pairing-based cryptography fall into this category. This algorithm has an asymptotic running time of $L_q\left(\frac{2}{3}, \left(\frac{64}{9} \cdot \frac{\lambda+1}{\lambda}\right)^{1/3}\right)$ for the medium characteristic case, and an asymptotic running time of $L_q(\frac{1}{3}, \sqrt[3]{\frac{32}{9}}) \approx L_q(\frac{1}{3}, 1.526)$ for the high characteristic case.

The work of Joux and Pierrot and the conjugation method of Barbulescu *et al.* both involve choosing polynomials f_1 and f_2 of degrees n and dn respectively. The sizes of the norms in the two number fields are kept balanced by making sure that all the coefficients of f_2 are very small and that all the coefficients of f_1 have absolute value at most $p^{1/d}$. More details about the algorithm of Joux and Pierrot is provided in Section 3.7.

For more information on the history of discrete logarithm algorithms, and an up-to-date comparison of the best known algorithms, see Chapter 4 of the paper by Joux, Odlyzko and Pierrot [29].

Given a particular DLP instance, it is difficult to determine a good estimate for the running time of the NFS. First of all, the running time estimates given above are all

Algorithm	Characteristic size	c
Joux	high	1.923
Barbulescu and Pierrot	high	1.902
Joux and Pierrot	high of special form	1.526
Joux	medium	2.423
Barbulescu and Pierrot	medium	2.240
Barbulescu	medium	2.201
Joux and Pierrot	medium of special form	$\left(\frac{64}{9} \cdot \frac{\lambda+1}{\lambda}\right)^{1/3}$

Table 3.1: Some recent NFS algorithms for the high and medium prime cases. The approximate running time of an algorithm is given by $L_q\left(\frac{1}{3}, c\right)$.

asymptotic, in the sense that the running time will approach the stated time only as $q \to \infty$. Some terms appearing in the running time analysis are routinely ignored on the grounds that they will become negligible as p goes to infinity. Second, for any prime power $q = p^n$ and any positive constant α, there exists a positive constant c such that $p = L_q(\alpha, c)$. As such, the distinction between the low, medium and high characteristic cases is not very well defined. Most running time analyses are only valid for relatively small c values, but exactly how small the values need to be is often not well understood. Finally, recall that proper use of L_q notation involves working with a $o(1)$ term, so specifying exact α and c values for a given instance does not really make sense theoretically.

To illustrate some of these issues, consider the case of BN curves. Recall from Section 2.2.1 that BN curves of cryptographic interest have embedding degree $k = 12$ and use 256-bit primes p. If we choose to write $q = p^{12} = L_q(\alpha, c)$ using $\alpha = \frac{1}{3}$, we get $c \approx 3.548$. Writing $p = L_q(\frac{2}{3}, c)$ gives a c value of ≈ 0.544. Both of these c values are small enough that the instance could be said to fall into the corresponding α case.

Joux *et al.* provide a plot that provides some guidelines about which of their two algorithms to use [28, Fig. 1]. Writing $p = L_q(\frac{2}{3}, c)$, they show how the running time of the NFS varies based on c. Based on the plot, $c = 2.5$ seems to be the approximate cutoff between the medium and high characteristic cases. As we shall see in Sections 3.5 and 3.6, both MNT and BN curves fall into the medium prime case based on this heuristic.

In general, although any prime power could be considered to fall under any case, we should choose the case that will produce the best running time. Our best bet to find a good algorithm is likely to compute the running times arising from each reasonable algorithm, and take whichever one is best. In Sections 3.5 to 3.7 we will check how different algorithms compare for particular DLP instances.

3.2 Choice of Polynomials

There are several choices for how to pick the irreducible polynomials $f_1, f_2 \in \mathbb{Z}[x]$ that will be used for the medium characteristic case of the NFS. The only necessary condition is that f_1 and f_2 have a common irreducible factor of degree n over $\mathbb{F}_p$, as mentioned in Section 2.3. The goal in choosing f_1 and f_2 is to maximize the probability that elements of the sieving space will have B-smooth norms.

Joux *et al.* [28] use a pair of polynomials of degree n of the form $f_2 = f_1 + p$, with f_1 being chosen to have very small coefficients. If γ is an upper bound on the absolute value of the coefficients of f_1, then $\gamma + p$ is an upper bound for f_2. Given an element $\alpha \in \mathcal{R}$, since $\deg f_1 = \deg f_2$, Equation (3.2) for the product of the norms of α reduces to

$$\begin{aligned} |N_{K_1}(\alpha)N_{K_2}(\alpha)| &\leq (n+t)!^2 S^{2n} \gamma^t (p+\gamma)^t \\ &= (n+t)!^2 S^{2n} p^{t+o(1)}. \end{aligned} \tag{3.3}$$

We justify our use of $o(1)$ notation here by saying that as p goes to infinity, we will be able to find a polynomial f_1 for which $\frac{\gamma}{p}$ approaches zero.

Benger *et al.* [9] use polynomials of the form $f_1 = x^n + a_2$ and $f_2 = a_1 x^n + i$, where a_1, a_2 are integers of size $\approx \sqrt{p}$ and i is a small integer such that $a_1 a_2 = p + i$. If we assume that as p goes to infinity, we can find a polynomial f_1 for which $\frac{i}{p}$ approaches zero, then we again get Equation (3.3) as a bound on the absolute value of the product of the norms of α.

A third choice of polynomials is suggested by Joux and Pierrot [30] for use with Barreto-Naehrig curves. They make use of the additional structure of the fields over which these curves are defined by working with polynomials f_1 and f_2 of degree 12 and 48 respectively. However, the bounds on the coefficients of f_1 and f_2 are more complicated then for the above two choices of polynomials. A different upper bound on the product of the norms of α is obtained. This bound and the resulting running time analysis will be covered in more detail in Section 3.7.

3.3 Estimating Smoothness Probability

To get a better estimate for the running time of the NFS, we wish to estimate the probability that an element $\alpha \in R$ has norms that are both B-smooth.

For $i = 1, 2$, let c_i be such that that $N_{K_i}(\alpha) \leq L_q(r, c_i)$ for all $\alpha \in R$. Let d be such that our sieving bound S satisfies $S = L_q(s, d)$. These assumptions will both be valid for the choice of parameters outlined in Section 3.4.

Under the same assumption as in the beginning of this chapter, namely that norms are just as likely to be B-smooth as are randomly chosen integers of the same size as the norm, we can apply Theorem 3.1 to get that the probability of $N_{K_i}(\alpha)$ being B-smooth is at least

$$L_q\left(r - s, -(r - s)\frac{c_i}{d}\right).$$

Note that this is only a lower bound on the probability, since if $N_{K_i}(\alpha)$ were significantly smaller than the upper bound of $L_q(r, c_i)$ than we would expect a higher probability of smoothness. We will see in Chapter 4 that most elements of the sieving space do in fact have norm much smaller than the upper bound.

Clearly, by unique factorization, the product of the norms is B-smooth if and only if both norms are B-smooth. Let $N_j = N_{K_i}(\alpha)$ for $i = 1, 2$. Then we have that

$$\begin{aligned}\Pr(N_1 N_2 \text{ is } B\text{-smooth}) &= \Pr(N_1 \text{ is } B\text{-smooth and } N_2 \text{ is } B\text{-smooth}) \\ &= \Pr(N_1 \text{ is } B\text{-smooth}) \cdot \Pr(N_2 \text{ is } B\text{-smooth} \mid N_1 \text{ is } B\text{-smooth}).\end{aligned}$$

If we further assume that the probability that N_2 is B-smooth is independent of the probability that N_1 is B-smooth, then we can write the probability of both being B-smooth as

$$\Pr(N_1 \text{ is } B\text{-smooth}) \cdot \Pr(N_2 \text{ is } B\text{-smooth}) \geq L_q\left(r - s, -(r - s)\frac{c_1 + c_2}{d}\right).$$

Since the probabilities are independent, this makes it possible to study the norm values separately when studying smoothness probability.

However, the above independence assumption may not actually be valid since the values of N_1 and N_2 are clearly dependent on each other, since they are both norms of the same element α. As such, it seems reasonable that the probabilities of the two norms being B-smooth are also not independent. Nonetheless, in this thesis we will always treat N_1 and N_2 as being independent.

Although we can only find lower bounds on the probability of smoothness, we will use these bounds to approximate the actual probability in our running time analyses in Sections 3.4 and 3.7.2.

3.4 Choice of Parameters

Our goal in this section is to choose a smoothness bound B, a sieving bound S and a sieving degree t to make sure that we can generate enough relations, while balancing the running times of the sieving step and the linear algebra step. To do so, we follow the procedure outlined by Joux *et al.* in their Appendix A.2 [28].

The analysis done in this section is only valid for the choice of polynomials used by Joux *et al.* and Benger *et al.* for the medium characteristic case. The polynomials used by Joux and Pierrot require a separate analysis which can be found in Section 3.7.

Let our sieving bound be given by

$$S = \exp\left(\frac{2c'}{t+1}(\log q)^{\frac{1}{3}}(\log\log q)^{\frac{2}{3}}\right) = L_q\left(\frac{1}{3}, \frac{2c'}{t+1}\right) \tag{3.4}$$

for some constant c' whose value we will determine later.

For $t < n$, the size of our sieving space is then

$$S^{t+1} = L_q\left(\frac{1}{3}, 2c'\right).$$

Writing $p = L_q(\frac{2}{3}, c)$ gives

$$\begin{aligned} \log p &= c(\log q)^{\frac{2}{3}}(\log\log q)^{\frac{1}{3}} \\ \iff \frac{\log q}{n} &= c(\log q)^{\frac{2}{3}}(\log\log q)^{\frac{1}{3}} \\ \iff n &= \frac{1}{c}\left(\frac{\log q}{\log\log q}\right)^{\frac{1}{3}}. \end{aligned} \tag{3.5}$$

Note that our choice of S gives

$$\begin{aligned} S^{2n}p^t &= \exp\left(\frac{4c'n}{t+1}(\log q)^{\frac{1}{3}}(\log\log q)^{\frac{2}{3}} + ct(\log q)^{\frac{2}{3}}(\log\log q)^{\frac{1}{3}}\right) \\ &= \exp\left(\left(\frac{4c'}{(t+1)c} + ct\right)(\log q)^{\frac{2}{3}}(\log\log q)^{\frac{1}{3}}\right). \end{aligned}$$

Using Equation (3.3) to upper bound the absolute value of the product of the norms, we get that

$$\begin{aligned} |N_{K_1}(\alpha)N_{K_2}(\alpha)| &\leq (n+t)!^2 p^{o(1)} \exp\left(\left(\frac{4c'}{(t+1)c} + ct\right)(\log q)^{\frac{2}{3}}(\log\log q)^{\frac{1}{3}}\right) \\ &= \exp\left(\left(\frac{4c'}{(t+1)c} + ct + o(1)\right)(\log q)^{\frac{2}{3}}(\log\log q)^{\frac{1}{3}}\right) \\ &= L_q\left(\frac{2}{3}, \frac{4c'}{(t+1)c} + ct\right), \end{aligned}$$

where we assume that $(n+t)!^2\frac{1}{q}$ approaches zero as q goes to infinity.

Let our smoothness bound be given by

$$B = L_q\left(\frac{1}{3}, c'\right). \tag{3.6}$$

Then the probability of both norms being B-smooth is

$$P \geq L_q\left(\frac{1}{3}, -\frac{1}{3}\left(\frac{4}{(t+1)c} + \frac{ct}{c'}\right)\right). \tag{3.7}$$

For the linear algebra step to be successful with high probability, we want to generate at least B relations. Thus we want that $S^{t+1}P = B$.

Note that the sieving stage takes time roughly equal to the size S of our sieving space $\mathcal{R}$ since we will likely end up having to test most of the elements in $\mathcal{R}$ for smoothness in order to find sufficient relations. As mentioned in Section 2.3, the linear algebra stage takes time $|\mathcal{B}|^2 \approx B^2$. To make both stages take a roughly equal amount of time, we thus want to set $S^{t+1} = B^2$. This will result in a total running time for the NFS of $B^2 = L_q\left(\frac{1}{3}, 2c'\right)$. Although we have only obtained a lower bound on P, we will use this bound as an estimate for P in order to balance the two stages.

To get $S^{t+1} = B^2$ we want to get $B = 1/P$. To do so, we need to choose c' such that

$$c' = \frac{1}{3}\left(\frac{4}{(t+1)c} + \frac{ct}{c'}\right).$$

The following method for determining c' is motivated by Joux *et al.*'s Appendix A.2 [28]. First we rewrite the above equation as

$$3c' - \frac{4}{(t+1)c} - \frac{ct}{c'} = 0.$$

Next solve for c':

$$c' = \frac{1}{3}\left(\frac{2}{(t+1)c} + \sqrt{\frac{4}{(t+1)^2c^2} + 3tc}\right). \tag{3.8}$$

We want to find the smallest possible value for c' given a particular value for p, which corresponds to a particular value of c. Thus we treat c as a constant and take the derivative of c' with respect to t. This is a different approach to that taken by Joux *et al.*, who are interested in finding the smallest value for c' amongst all possible values for p and hence take the derivative with respect to c. Taking the derivative gives

$$\begin{aligned}
\frac{dc'}{dt} &= \frac{-1}{3}\left(\frac{2}{(t+1)^2c} + \frac{\frac{8}{(t+1)^3c^2} - 3c}{2\sqrt{\frac{4}{(t+1)^2c^2} + 3tc}}\right) = 0 \\
&\iff \\
\frac{2}{(t+1)^2c}\left(2\sqrt{\frac{4}{(t+1)^2c^2} + 3tc}\right) &= -\frac{8}{(t+1)^3c^2} + 3c \\
&\iff \\
\frac{16}{(t+1)^4c^2}\left(\frac{4}{(t+1)^2c^2} + 3tc\right) &= \frac{64}{(t+1)^6c^4} - \frac{48}{(t+1)^3c} + 9c^2 \\
&\iff \\
\frac{64}{(t+1)^2c^2} + 48tc &= \frac{64}{(t+1)^2c^2} - 48(t+1)c + 9t^2(t+1)^4c^4 \\
&\iff \\
48(2t+1)tc &= 9(t+1)^4c^4 \\
&\iff \\
0 &= 3(t+1)^4c^3 - 16(2t+1). \qquad (3.9)
\end{aligned}$$

Since our sieving degree t must be an integer, we should let t be the positive integer closest to the real root of $3(t+1)^4c^3 - 16(2t+1) = 0$ in order to minimize c'.

3.5 Parameters for MNT Curves

We now turn towards finding specific parameter values for some NFS instances of cryptographic interest. Recall from Section 2.2.1 that MNT curves are often used at the 80-bit security level. The security of the ECDLP for these curves reduces to the DLP in $\mathbb{F}_{p^n}^*$ for a 160-bit prime p and $n = 6$. We used the computational algebra system Magma [12] to calculate parameter values.

From Equation (3.5), we get

$$c = \frac{1}{n}\left(\frac{\log q}{\log\log q}\right)^{\frac{1}{3}} \approx 0.780.$$

Substituting the exact value for c into Equation (3.9) and solving for t gives ≈ 1.632 as the real root, so we let $t = 2$. We next use Equation (3.8) to determine the value of c', namely

$$c' \approx 1.060.$$

Choosing our smoothness bound and sieving bound as in Equations (3.6) and (3.4), gives

$$B = L_q\left(\frac{1}{3}, c'\right) \approx 1.005 \times 10^{14} \approx 2^{46.5},$$

and

$$S = L_q\left(\frac{1}{3}, \frac{2c'}{t+1}\right) \approx 2.162 \times 10^{9} \approx 2^{31.0}.$$

With these parameters, the running time of the NFS is

$$L_q\left(\frac{1}{3}, 2c'\right) \approx L_q\left(\frac{1}{3}, 2.120\right) \approx 1.010 \times 10^{28} \approx 2^{93.0},$$

which, as expected, is computationally infeasible. More importantly, the running time is greater than 2^{80} so MNT curves still appear to offer the desired 80-bits of security.

Benger *et al.* [9] use the unmodified analysis of Joux *et al.* to find suitable parameters for MNT curves providing 80-bits of security. They use a prime

$$p = 1461501637330902918203684832716283019655932543333,$$

a sieving bound of $S = 2075482890$ and a sieving degree of $t = 2$. Plugging these parameters into Equation (3.8) gives a running time of $L_q\left(\frac{1}{3}, 2c'\right)$ with $2c' \approx 2.400$. This is worse than the running time obtained using our parameters, suggesting that our analysis does indeed provide a better set of parameters for the 80-bit security case. Indeed, comparing our results to Table (3.1), we see that our parameters provide a better running time than any of the general purpose NFS algorithms for the medium prime case.

3.6 Parameters for BN Curves

As mentioned in Section 2.2.1, BN curves are often used at the 128-bit security level. This corresponds to solving the DLP in a field of size p^n where p is a 256-bit prime and $n = 12$. We give specific parameters to use for this DLP setting, the values of which were calculated using Magma [12].

From Equation (3.5), we get

$$c = \frac{1}{n}\left(\frac{\log q}{\log\log q}\right)^{\frac{1}{3}} \approx 0.544.$$

Substituting the exact value for c into Equation (3.9) and solving for t gives $t \approx 2.865$ as the real root, so we let $t = 3$. We next use Equation (3.8) to determine the value of c', namely

$$c' \approx 1.105.$$

Choosing our smoothness bound and sieving bound as in Equations (3.6) and (3.4), gives

$$B = L_q\left(\frac{1}{3}, c'\right) \approx 9.999 \times 10^{23},$$

and

$$S = L_q\left(\frac{1}{3}, \frac{2c'}{t+1}\right) \approx 9.999 \times 10^{11}.$$

These parameters give a running time for the NFS of

$$L_q\left(\frac{1}{3}, 2c'\right) \approx L_q\left(\frac{1}{3}, 2.210\right) \approx 9.998 \times 10^{47} \approx 2^{159.5},$$

which is not only computationally infeasible, but is larger than 2^{128}, suggesting that BN curves still offer 128-bits of security.

Benger *et al.* [9] also use the unmodified analysis of Joux *et al.* to find a suitable sieving degree of $t = 3$ or $t = 4$ for BN curves providing 128-bits of security, but do not provide any sieving bound data for comparison.

Looking at Table 3.1, this is actually worse than the result of Barbulescu *et al.* [4]. We thus turn to the results of Joux and Pierrot [30] for a better choice of polynomials to use with BN curves. This will be the focus of the rest of the chapter.

3.7 Joux-Pierrot Polynomials for BN Curves

As mentioned in Section 3.1, Joux and Pierrot [30] developed an alternate version of the NFS that has a better running time than the version proposed by Joux *et al.*. Their version is only valid for finite fields whose characteristic p can be written as $p = P(u)$ for some low degree polynomial $P \in \mathbb{Z}[x]$ with small coefficients. They propose the use of special polynomials f_1 and f_2 that make use of the special form of the characteristic. Fortunately, this method can be used for BN curves. As was seen in Section 2.2.1, BN curves of cryptographic interest are elliptic curves over $\mathbb{F}_p$, where $p = P(u) = 36u^4+36u^3+24u^2+6u+1$ is a 256-bit prime and the embedding degree is $n = 12$.

Joux and Pierrot suggest using $f_1(x) = x^{12} + r(x) - u$, where $r(x)$ is a polynomial of degree < 12, preferably of much smaller degree, all of whose coefficients are -1, 0 or 1. The second polynomial is then chosen as $f_2(x) = P(x^{12} + r(x)) = P(f_1(x) + u)$, with degree $\deg P \deg f_1 = 4 \cdot 12 = 48$. It was shown by Joux and Pierrot that f_2 is a multiple of f_1, so they have a common irreducible factor of degree n over $\mathbb{F}_p$ and hence are a valid choice of polynomials for the NFS.

3.7.1 Choosing $r(x)$

Before we attempt to determine the running time and find good parameters for BN curves using Joux-Pierrot polynomial, we first attempt to determine exactly how low of a degree we will be able to use for $r(x)$.

The number N of irreducible monic polynomials of degree k over $\mathbb{F}_p$ is bounded by

$$\frac{1}{k}(p^k - p) \leq N \leq \frac{1}{k}\left(p^k - \frac{p}{p-1}(p^{k/2} - 1)\right) \tag{3.10}$$

[32, Exercises 3.26 and 3.27]). Thus for sufficiently large p, the probability of a randomly chosen polynomial of degree k being irreducible over $\mathbb{F}_p$ is about $1/k$. Joux and Pierrot suggest that we should be able to find $r(x)$ of degree ≤ 2 such that $f_1(x) = x^{12} + r(x) - u$ is irreducible. In fact, we would expect there to be about two such choices for $r(x)$, since there are three choices for each coefficient of $r(x)$, giving $3^3 \geq 2 \cdot k = 24$ choices for $r(x)$.

However, it turns out that although the theoretical average number of valid choices for $r(x)$ of degree ≤ 2 is around two, there are quite often no choices that work. This is because for most choices for $r(x)$ there is another choice $s(x)$ such that $x^{12} + r(x) - u$ is irreducible over $\mathbb{F}_p$ if and only if $x^2 + s(x) - u$ is too. All the pairs are related by multiplying one of their coefficients by -1. Specifically, there are twelve disjoint pairs:

$$\begin{array}{llll}
x^2 \pm x + 1, & -x^2 \pm x + 1, & x^2, \pm x - 1 & -x^2 \pm x - 1, \\
x^2 \pm x, & -x^2 \pm x, & \pm x + 1, & \pm x - 1, \\
\pm x, & \pm x^2 + 1, & \pm x^2 - 1, & \pm x^2.
\end{array}$$

Having more dependencies between polynomials lowers the probability of their being one which corresponds to an irreducible $f_1(x)$. If we assume that the only dependencies among $r(x)$ choices are the twelve pairs above, then the probability of no polynomial of degree ≤ 2 giving rise to an irreducible is simply the probability that $r(x) = -1, 0, 1$ each do not work, and that none of the twelve pairs work. From Equation (3.10), for any given polynomial $r(x)$ of degree < 12, we can approximate the probability of $f_1(x) = x^{12} + r(x) - u$ being irreducible by $\frac{1}{12}$. The probability of no polynomial of degree ≤ 2 working is thus $(1 - \frac{1}{12})^{12+3} \approx 0.27$. Thus we would expect that for about a quarter of u values, there will be no $r(x)$ of degree ≤ 2 such that $f_1(x) = x^{12} + r(x) - u$ is irreducible.

To see why we get this pattern of pairs, let $g \in \mathbb{F}_p[x]$. Let $g^* \in \mathbb{F}_p[x]$ be obtained from g by negating all the odd degree terms:

$$[x^i]g^* = \begin{cases} [x^i]g & \text{if } i \text{ is even} \\ -[x^i]g & \text{if } i \text{ is odd} \end{cases}.$$

Suppose that there exist non-constant polynomials $u, v \in \mathbb{F}_p[x]$ such that $g = uv$. Then $u^*v^* = (uv)^* = g^*$. Thus g is irreducible if and only if g^* is irreducible.

Nine of the twelve pairs above give a pair of polynomials f_1 of the form g, g^*. The remaining three pairs are $\pm x^2, \pm x^2 + 1$ and $\pm x^2 - 1$. We were unable to find a complete theoretical explanation for the relationship between these pairs, but we do have some experimental evidence. For 500 randomly selected u values that produce a 256-bit prime p, we used Magma [12] to test whether each of the six polynomials produce an irreducible f_1 polynomial. For each u, one member of the pair gave an irreducible f_1 if and only if the other member of the pair did too. Additionally, given a pair r_1, r_2, if we can write

$x^{12} + r_1 - u = ab$ for polynomials a, b having only even degree terms, we can obtain a factorization for $x^{12} + r_2 - u$ as follows. Given a polynomial $g \in \mathbb{F}_p[x]$, let $\hat{g} \in \mathbb{F}_p[x]$ be obtained from g by flipping the coefficient of x^i if and only if $i \equiv 2 \pmod 4$:

$$[x^i]\hat{g} = \begin{cases} [x^i]g & \text{if } i \not\equiv 2 \pmod 4 \\ -[x^i]g & \text{if } i \equiv 2 \pmod 4 \end{cases}.$$

Note that for each of the above pairs, the polynomials f_1 corresponding to the pair are of the form $g, \hat{g}$. We thus get that $\hat{a}\hat{b} = \hat{ab} = x^{12} + r_2 - u$, where the first equality follow from the fact that there are no odd degree terms in either a or b.

Since we expect that for roughly a quarter of u values there are no polynomials $r(x)$ of degree ≤ 2 that produce an irreducible f_1, we will have to look at cubic $r(x)$ choices. A similar pattern of pairs occurs for cubics, which results in 27 pairs, each of the form g, g^*. The pairs are listed in Appendix A.1. Assuming that no other dependencies exist among the cubic $r(x)$ choices, we expect that for most u values, there will be an $r(x)$ of degree ≤ 3 that works. Using Equation (3.10) again, the probability of no such $r(x)$ working is approximately $(1 - \frac{1}{12})^{3+12+27} \approx 0.026$.

Joux and Pierrot also briefly discuss the use of $r(x)$ choices of higher degree. Although we will not consider high degree $r(x)$ choices in this thesis, they can be used in the NFS without affecting its asymptotic running time as long as $f_2 = P(x^{12} + r(x))$ has small coefficients.

3.7.2 Finding Optimal Parameters

We now follow the calculations done in Joux and Pierrot's Section 6.2 [30] to find a running time estimate for the NFS, in terms of the sieving degree t for Joux-Pierrot polynomials. We will then use Magma [12] to determine the best value of t for BN curves providing 128-bits of security, and give specific values for the corresponding smoothness and sieving bounds.

Note that Joux-Pierrot polynomials have smaller upper bounds on the absolute values of their coefficients then the polynomials used for the analysis in Section 3.4. Although f_2 has a higher degree than in Section 3.4, we will see that we still end up with smaller bound on the absolute value of the product of the norms and hence a better running time for the NFS.

Recall from Equation (3.1) that given an element $\alpha \in \mathcal{R}$, we can bound the absolute value of the norm by $|N_{K_i}(\alpha)| \leq (n_i + t)!\, S^{n_i} D_i^t$, where $n_i = \deg f_i$, S is our sieving bound and D_i is an upper bound on the absolute values of the coefficients of f_i, for $i = 1, 2$. The absolute values of the coefficients of f_1 are bounded by $u + 1 = O(p^{\frac{1}{4}})$.

An upper bound for the absolute values of the coefficients of $f_2 = P(x^{12} + r(x))$ is slightly harder to find. We will first find, for any positive integer k, an upper bound for the absolute values of the coefficients of $(x^{12} + r(x))^k$. Let $r'(x)$ be obtained from $r(x)$ by taking the absolute values of all its coefficients. Note that an upper bound on the coefficients of $(x^{12} + r'(x))^k$ is also a valid upper bound on the absolute values of the coefficients of $(x^{12} + r(x))^k$. We can find the sum of all the coefficients of $(x^{12} + r(x))^k$ by plugging in $x = 1$. So $(1 + r(1))^k = (2 + \deg r)^k$ is an upper bound on the absolute values of the coefficients of $(x^{12} + r(x))^k$.

We can now easily find an upper bound on the absolute values of the coefficients of f_2. Since

$$f_2 = P(x^{12} + r) = 36(x^{12} + r)^4 + 36(x^{12} + r)^3 + 24(x^{12} + r)^2 + 6(x^{12} + r) + 1,$$

we get a bound of

$$\begin{aligned} &36(2 + \deg r)^4 + 36(2 + \deg r)^3 + 24(2 + \deg r)^2 + 6(2 + \deg r) + 1 \\ \leq\, &5 \cdot 36(2 + \deg r)^4. \end{aligned} \tag{3.11}$$

Note that this is actually a bound on the sum of the absolute values of the coefficients of f_2, a fact that we will use in Chapter 6 to find better norm bounds.

Joux and Pierrot noted that if $\deg r = O(\log(n_1))$, then we get a bound for f_2 of $O(\log(n_1)^4)$ [30, Section 5.2]. Since $n_2 = 4n_1$, we can upper bound the absolute value of the product of the norms of α by

$$\begin{aligned} |N_{K_1}(\alpha) N_{K_2}(\alpha)| &\leq \big((n_1 + t)!\, S^{n_1}\, (u + 1)^t\big) \big((4n_1 + t)!\, S^{4n_1} \cdot (180(2 + \deg r)^4)^t\big) && (3.12) \\ &\leq \Big((n_1 + t)!\, S^{n_1}\, O(p^{\frac{t}{4}})\Big) \big((4n_1 + t)!\, S^{4n_1}\, O(\log n_1)^{4t}\big) \\ &\leq (n_1 + t)!\, (4n_1 + t)!\, S^{5n_1} p^{\frac{t}{4} + o(1)} (\log n_1)^{4t + o(1)}. && (3.13) \end{aligned}$$

We use the same sieving bound S and smoothness bound B as in Equations (3.4) and (3.6). Using Equation (3.5) to write p and n_1 in terms of c and q, we get

$$\begin{aligned} S^{5n_1} p^{\frac{t}{4}} &= \exp\left(\frac{10 n_1 c'}{t + 1} (\log q)^{\frac{1}{3}} (\log\log q)^{\frac{2}{3}} + \frac{ct}{4} (\log q)^{\frac{2}{3}} (\log\log q)^{\frac{1}{3}}\right) \\ &= \exp\left(\left(\frac{10c'}{(t + 1)c} + \frac{ct}{4}\right) (\log q)^{\frac{2}{3}} (\log\log q)^{\frac{1}{3}}\right) \\ &= L_q\left(\frac{2}{3}, \frac{10c'}{(t + 1)c} + \frac{ct}{4}\right). \end{aligned}$$

We then bound the absolute value of the product of the norms of α by

$$\begin{aligned}|N_{K_1}(\alpha)N_{K_2}(\alpha)| &\le (n_1+t)!\,(4n_1+t)!\,(\log n_1)^{4t+o(1)}p^{o(1)}L_q\left(\frac{2}{3},\frac{10c'}{(t+1)c}+\frac{ct}{4}\right)\\ &= L_q\left(\frac{2}{3},\frac{10c'}{(t+1)c}+\frac{ct}{4}\right),\end{aligned}$$

where we assume that $(n_1+t)!\,(4n_1+t)!\,\log(n_1)^{4t+o(1)}p^{o(1)}\cdot\frac{1}{q}$ approaches zero as q goes to infinity.

The probability of both norms being B-smooth is then

$$P \ge L_q\left(\frac{1}{3},-\frac{1}{3}\left(\frac{10}{(t+1)c}+\frac{ct}{4c'}\right)\right). \tag{3.14}$$

We now repeat the calculations done in Section 3.4 in order to find the optimal values of t and c'. We again use our lower bound on P as an estimate for P in order to balance the sieving and linear algebra stages. We start by setting $B = 1/P$ so that

$$\begin{aligned}c' &= \frac{1}{3}\left(\frac{10}{(t+1)c}+\frac{ct}{4c'}\right)\\ &\iff\\ 0 &= 3c'^2 - \frac{10c'}{(t+1)c} - \frac{tc}{4}\\ &\iff\\ c' &= \frac{1}{6}\left(\frac{10}{(t+1)c}+\sqrt{\frac{100}{(t+1)^2c^2}+3tc}\right)\\ &= \frac{1}{3}\left(\frac{5}{(t+1)c}+\sqrt{\frac{25}{(t+1)^2c^2}+\frac{3tc}{4}}\right).\end{aligned} \tag{3.15}$$

Note that this corresponds to Equation (7) in Joux and Pierrot's paper.

We will end up with a running time of $L_q(\frac{1}{3},2c')$, so we want to minimize the value of c' in the above equation. Taking the derivative of c' with respect to t, we have

$$\frac{dc'}{dt} = \frac{1}{3}\left(\frac{-5}{(t+1)^2c} + \frac{\frac{-50}{(t+1)^3c^2}+\frac{3c}{4}}{2\sqrt{\frac{25}{(t+1)^2c^2}+\frac{3tc}{4}}}\right) = 0$$

$$\Longleftrightarrow$$

$$\frac{10}{(t+1)^2c}\sqrt{\frac{25}{(t+1)^2c^2}+\frac{3tc}{4}} = \frac{-50}{(t+1)^3c^2}+\frac{3c}{4}$$

$$\Longleftrightarrow$$

$$\frac{100}{(t+1)^4c^2}\left(\frac{25}{(t+1)^2c^2}+\frac{3tc}{4}\right) = \frac{2500}{(t+1)^6c^4} - \frac{75}{(t+1)^3c} + \frac{9c^2}{16}$$

$$\Longleftrightarrow$$

$$\frac{2500}{(t+1)^2c^2} + 75tc = \frac{2500}{(t+1)^2c^2} - 75(t+1)c + \frac{9(t+1)^4c^4}{16}$$

$$\Longleftrightarrow$$

$$400(2t+1) = 3(t+1)^4c^3. \tag{3.16}$$

Thus we should let t be the integer closest to the positive root of $3(t+1)^4c^3 - 400(2t+1) = 0$.

We can write c in terms of n_1 and q in the same way as in Equation (3.5) to get $c = \frac{1}{n_1}\left(\frac{\log q}{\log\log q}\right)^{\frac{1}{3}} \approx 0.544$. With this value of c, the real root of Equation (3.16) is then ≈ 10.665, so we let $t = 11$.

From Equation (3.15) we now get that $c' \approx 1.006$. Using Equations (3.6) and (3.4) to obtain smoothness and sieving bounds, we get $B \approx 7.129 \times 10^{21}$ and $S \approx 4387$.

These parameters give a running time for the NFS of

$$L_q\left(\frac{1}{3}, 2c'\right) \approx L_q\left(\frac{1}{3}, 2.012\right) \approx 5.081 \times 10^{43} \approx 2^{145.2},$$

which is greater than 2^{128}, showing that BN curves still offer 128-bits of security. As expected, these parameters give a better running time for BN curves than the choice of parameters given in Section 3.6, showing that Joux-Pierrot polynomials do actually offer an improvement. However, the running time is still computationally infeasible.

Comparing Equation (3.14) for smoothness probability with Equation (3.7) from Section 3.4, we see that for a fixed c value, the probability for Joux-Pierrot polynomials grows less quickly with respect to t. Thus we can work with a higher t value while still expecting that approximately the same percent of elements in the sieving space will produce a relation.

Recall from the definition of t in Section 2.3.3 that we need $t < \min(n_1, n_2)$ in order to get a sieving space of size S^{t+1}. Since we are working with $n_1 = 12$, $t = 11$ is actually the largest value of t for which our analysis holds. For larger t values, the size of the sieving space will be less than S^{t+1}, causing the sieving and linear algebra stages to no longer be balanced in terms of time.

Chapter 4

Experimental Results

In this section we collect data on the distribution of the norms of elements in the sieving space. Using this data, we look at how different NFS parameters affect the distribution and give some suggestions on how to pick a good choice of Joux-Pierrot polynomials.

We start by replicating the results on MNT norms found in the paper by Benger *et al.*. We then apply a similar process to BN curves, looking at the affects of $r(x)$ and u on the distribution.

Note that throughout this chapter, when we refer to the norm of an element we are actually talking about the absolute value of the norm of the element.

4.1 Methodology and Replicating BCC Results

All of the experiments done in this chapter involve picking a pair of number fields, K_1 and K_2, and calculating norms for randomly selected elements of the sieving space. Recall from Section 2.3.3 that our sieving space $\mathcal{R}$ is the set of all polynomials of the form $\sum_{j=0}^{t} a_j x^j \in \mathbb{Z}[x]$ with coefficients a_j satisfying $0 \leq a_j < S$ for $j = 0, \ldots, t$. To collect data on the sizes of norms, we used a C program based on a library by Charlemagne [15]. The program makes use of several other publicly available C libraries, namely GMP [24], GMP-ECM [54], FLINT [26], MPFR [21] and MPIR [25]. The computational algebra system Magma [12] was used to calculate primes p of the proper size, and to find irreducible polynomials f_1 and f_2 such that f_1 is also irreducible over $\mathbb{F}_p$. We repeatedly selected a random element $\alpha \in \mathcal{R}$ and calculated the absolute value of its norms $N_{K_i}(\alpha)$ for $i = 1, 2$. Sometimes the product of the norms was recorded, and sometimes the norms were recorded individually, depending on the goal of the particular experiment. For each experiment, we calculated norm values for 500000 elements of the sieving space.

Our initial experiment involved attempting to replicate the findings of Benger *et al.* for the 80-bit security MNT setting, in order to provide some validation of our results. In particular we worked with a 160-bit prime p, two degree $n = 6$ extensions K_1 and K_2 of $\mathbb{Q}$, and a value of $t = 2$ for the sieving space. We used the 160-bit prime

$$p = 1461501637330902918203684832716283019655932543333$$

found in their Appendix A. The corresponding smoothness bound and sieving bound are $B = 100505868921572$ and $S = 2161694322$, as calculated in Section 3.5. We followed the polynomial selection method described in Section 3.2 of their paper to generate our two extensions. By choosing random a_1 values between $\lfloor\sqrt{p}\rfloor - 10000$ and $\lfloor\sqrt{p}\rfloor$, and setting $a_2 = \lfloor p/a_1 \rfloor$, we found that $a_1 = 1208925819614629174706174$ and $a_2 = 1208925819614629174706178$ produced polynomials $f_1 = x^6 + a_2$ and $f_2 = x^6 + a_1 - 361$ that are irreducible over $\mathbb{F}_p$.

We calculated and recorded both norms for 500000 elements of the sieving space. The maximum product M was found to be approximately 2.181×10^{208}. This is significantly lower than our previously stated theoretical maximum from Equation (3.3), which works out to

$$(n+t)!^2 S^{2n} p^t = 3.616 \times 10^{217}$$

with the above choice of parameters (we have dropped the $p^{o(1)}$ factor in order to get an exact number).

To compare our results with those of Benger *et al.*, we started by creating a plot of the cumulative probability of the absolute value of the K_1 norm, $|N_{K_1}(\alpha)|$, as a fraction of the maximum observed K_1 norm. This plot can be seen in Figure (4.1). We chose to use only the first norm instead of the product of the norms in order to replicate Benger *et al.*'s Figure A.1. The validity of this choice is justified by our analysis in Section 3.3. The maximum observed K_1 norm M_1 was approximately 1.491×10^{105}.

The plot was created using the statistical computing package R [44], as were all other plots included in this thesis. For each norm $|N_{K_1}(\alpha)|$, we first calculated $|N_{K_1}(\alpha)| / M_1$ and then placed the norm into one of the 1000 buckets

$$\{((i-1)/1000, i/1000] : 1 \leq i \leq 1000, i \in \mathbb{Z}\}$$

between 0 and 1. The plot was then created based on the number of elements in each bucket. Note that similar plots for the K_1 and K_2 norms of BN curves are given in Section 5.1.

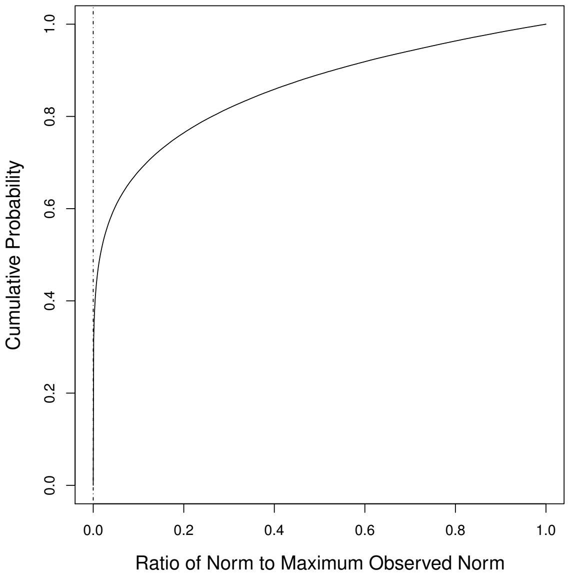

Figure 4.1: Plot of the cumulative probability that the MNT K_1 norm of a random element of the sieving space is at most some fraction of the maximum observed norm $M_1 \approx 1.491 \times 10^{105}$. Based on 500000 norm values, for $p = 1461501637330902918203684832716283019655932543333$ and $K_1 = \mathbb{Q}[x]/(x^6 + 1208925819614629174706178)$.

We can see from the plot that the majority of elements of the sieving space have K_1 norm much smaller than the maximum observed K_1 norm. We will shortly present more exact numbers, but a quick glance at the plot shows that at least half of the norms are less than $0.05M_1$. Thus, even if we were able to obtain a much better theoretical upper bound on the norm, we would still expect that most elements have norm much smaller than the theoretical bound.

It is worth noting that the largest observed absolute value of the K_2 norm, $|N_{K_2}(\alpha)|$, was $M_2 \approx 1.49130 \times 10^{105}$. This is very close to M_1 and supports our use of the same theoretical upper bound for both norms in Section 3.2. However, the fact that M is several bits larger than the product $M_1 M_2$ suggests that the use of upper bounds for each norm may not yield a tight upper bound on the product of the norms.

To get a better idea of how the distribution of norms behaves for norms much less than M_1, we calculated the percentage of elements with norm less than $10^i M_1$ for $i = -9, -8, \ldots, 0$. These values are given in Table 4.1. They show that the smallest 1% of norms are smaller than $10^{-9} M_1$.

Fraction of max	10^{-9}	10^{-8}	10^{-7}	10^{-6}	10^{-5}	10^{-4}	10^{-3}	10^{-2}	10^{-1}	10^{0}
Percentage of norms	1.78	3.81	6.78	10.0	14.7	21.6	31.6	46.5	68.2	100

Table 4.1: Percentage of MNT K_1 norms below various fractions of the maximum observed norm M_1 (cf. Figure 4.1).

We also calculated various percentiles for the MNT K_1 data, which are given in Table 4.2. From this data, we see that half of all tested elements in the sieving space have a norm less than $0.16M_1$.

Percentage of norms	10%	25%	50%	75%	90%	100%
Percentile	9.99×10^{-7}	0.000246	0.0158	0.178	0.530	1

Table 4.2: Percentiles for MNT K_1 norms. The percentile represents the percent of M_1 below which the given fraction of norms will fall (cf. Figure 4.1).

Our data on the distribution of MNT norms agrees with the data given by Benger *et al.*. Having found some support for our methodology, our next goal is to calculate and analyze similar data for BN curves.

4.2 Values of u and $r(x)$ for Use with BN Curves

We now start to gather some data on the distribution of norms for the 128-bit security BN setting, choosing polynomials according to the selection method of Joux and Pierrot [30] described in Section 3.7. We work with a 256-bit prime p, an extension K_1 of degree 12 and an extension K_2 of degree 48. For the sieving space, we use a sieving degree of $t = 11$ and a sieving bound of $S = 4387$, as calculated in Section 3.7.2.

Recall that the primes p used in the BN setting are chosen by picking a random integer u and such that $p = P(u)$ is a 256-bit prime, where $P(z) = 36z^4 + 36z^3 + 24z^2 + 6z + 1$. We let $f_1 = x^{12} + r(x) - u$ and $f_2 = P(x^{12} + r(x))$, where $r(x)$ is a polynomial of low degree all of whose coefficients are 1, -1 or 0. We will consider both randomly selected u values, as well as the value

$$u_1 = 6917529027641089837 = \text{0x6000000000001}F2D,$$

which allows for more efficient implementations of pairing-based cryptography. u_1 was shown by Devegili, Scott and Dahab to speed up the Ate pairing due to its low Hamming weight [18]. The corresponding prime $P(u_1)$ is

$$82434016654300679721217353503190038836571781811386228921167322412819029493183.$$

The purpose of our first BN experiments is to see how the choice of $r(x)$ can affect the distribution of norms, when u is held constant. In order to get a relation, we need both norms to be smooth and so we need to look at the distributions of both norms. Whereas the polynomials f_1 and f_2 that we used in Section 4.1 were of roughly the same form, having the same degree and the same size coefficients, this is not the case for the Joux-Pierrot polynomials. As such, we cannot assume that both norms will have the same distributions and so we choose to look at the product of the two norms.

Since we are interested in comparing u and $r(x)$ values to see which are most likely to give the most relations, we must develop a way to experimentally determine which of two norm distributions is best. There are several options for comparing distributions, some of which are more useful than others. One simply way to compare distributions is to look at their maximum observed norm values. Since maximum theoretical bounds are often used to compute theoretical running times for the NFS, this might seem like a natural way to compare distributions. However, this is quite clearly not a good method of comparison, as the maximum observed norm values are likely to change if an experiment is repeated.

In fact, any information regarding the size of the largest norm is of rather limited value. Even if we were able to calculate an exact upper bound on the norm, this information would not be sufficient to estimate the number of relations that we can expect to find. Since we do not need every element of the sieving space to give rise to a relation, the size of the largest norms is mostly insignificant.

A better method of comparing two distributions is to look at upper bounds that apply to some large portion of the norms. For example, by giving the 25th and 50th percentiles, we know that a quarter of elements have norms between those two values. This then allows us to use Theorem 3.1 to give approximate upper and lower bounds for the probability that each of these elements is smooth.

We will thus give several percentiles for each distribution that we calculate. We also provide a plot of the distributions, which allows us to graphically compare percentiles. Due to the large sizes of the numbers involved, we will actually work with base-10 logarithms of products of norms, instead of the norm products themselves. We use base-10 instead of binary to further reduce the size of the numbers.

4.2.1 $r(x)$ Choices for u_1

For our first BN experiment, we will look at different $r(x)$ values for the specially chosen u value u_1. As was mentioned in Section 3.7.1, we expect that for roughly one quarter of u values, a cubic $r(x)$ is the lowest degree polynomial that will give a value of f_1 which is irreducible over $\mathbb{F}_p$. For u_1 we are in this unfortunate situation. In fact, there are only two cubic $r(x)$ choices that work for u_1, namely $r(x) = \pm x^3$. We compare both possible $r(x)$ choices for u_1. As in Section 4.1, for each choice of $r(x)$ we calculated the norms for 500000 randomly chosen elements of the sieving space. However, for the reasons discussed above, we will this time look at the product of their norms, rather than just the K_1 norm.

In Figure 4.2, we plot the cumulative probability that the base-10 logarithm of the product of the norms is at most some amount. Selected percentiles are given in Table 4.3, such as the 50% percentile, which states that, for $r(x) = x^3$, half of the values calculated for the norm product were less than $10^{437.2}$. It is clear from both the plot and the table of percentiles that $r(x) = x^3$ is a better choice than $-x^3$.

Although the maximum observed product of norms is not very useful in comparing distributions, it is interesting to see how it compares with our theoretical upper bound. Using Equation (3.12), we get a theoretical maximum norm product for u_1 of 7.704×10^{583}. The largest observed norm products for $r(x) = x^3$ and $-x^3$ were about 6.820×10^{454} and 7.990×10^{456}, respectively. Although it is possible that there could be an element of the sieving space with norm significantly larger than any of the observed norms, it appears as

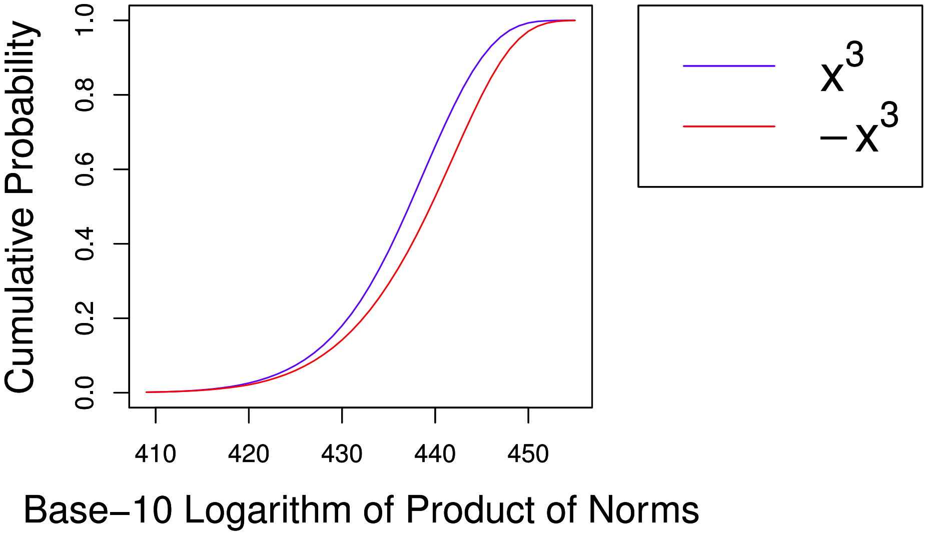

Figure 4.2: Plot of the cumulative probability that the base-10 logarithm of the BN product of the norms of a random element of the sieving space is at most some amount. Based on 500000 elements using $u_1 = 6917529027641089837$.

if Equation (3.12) does not give a very tight bound. In Chapter 6, we will attempt to give a better theoretical upper bound.

Percentage of norms		10%	25%	50%	75%	90%	100%
$r(x)$	x^3	426.6	432.1	437.2	441.6	445.0	454.8
	$-x^3$	427.9	433.8	439.5	444.0	447.3	456.9

Table 4.3: Percentiles for BN product of norms for u_1. The percentile represents the base-10 logarithm of the product of the norms below which the given percent of norms fall (cf. Figure 4.2).

The maximum observed K_1 and K_2 norms for $r(x) = x^3$ were about 8.796×10^{250} and 1.328×10^{204}, respectively. For $r(x) = -x^3$, the maximum norms were about 8.796×10^{250} and 2.598×10^{206}. In Chapter 6, we will see that the choice of $r(x)$ has very little effect on the theoretical upper bound for the K_1 norm, explaining why the maximum values were nearly identical.

4.2.2 $r(x)$ Choices of Different Degrees

Recall that our choice of $r(x)$ can be any polynomial of degree less than twelve whose coefficients are all either 1, -1 or 0, and for which $f_1(x) = x^{12} + r(x) - u$ is irreducible over $\mathbb{F}_p$. In this section we attempt to determine what effect, if any, the degree of $r(x)$ has on the distribution of the product of the norms. We also consider how the number of terms in $r(x)$ can affect the distribution.

To do so, we searched for a single u value for which there were polynomials $r(x)$ of different degrees, each of which produced an irreducible polynomial f_1. We found that the value

$$u = 6790547177159395210$$

has 19 $r(x)$ values of degree at most 3 that work, including $r(x) = 0, x^2 + x + 1, x^3$ and $x^3 + x^2 - x - 1$. The polynomials f_2 corresponding to these four choices of $r(x)$ have 5, 25, 14 and 33 terms respectively. Their coefficients with largest absolute values are 36, 1236, 216 and 1044 respectively.

We again calculated the norms for 500000 randomly chosen elements of the sieving space, for each choice of $r(x)$. A plot comparing the cumulative probability that the base-10 logarithm of the product of the norms is at most some amount is given in Figure 4.3. Some corresponding percentile values are given in Table 4.4.

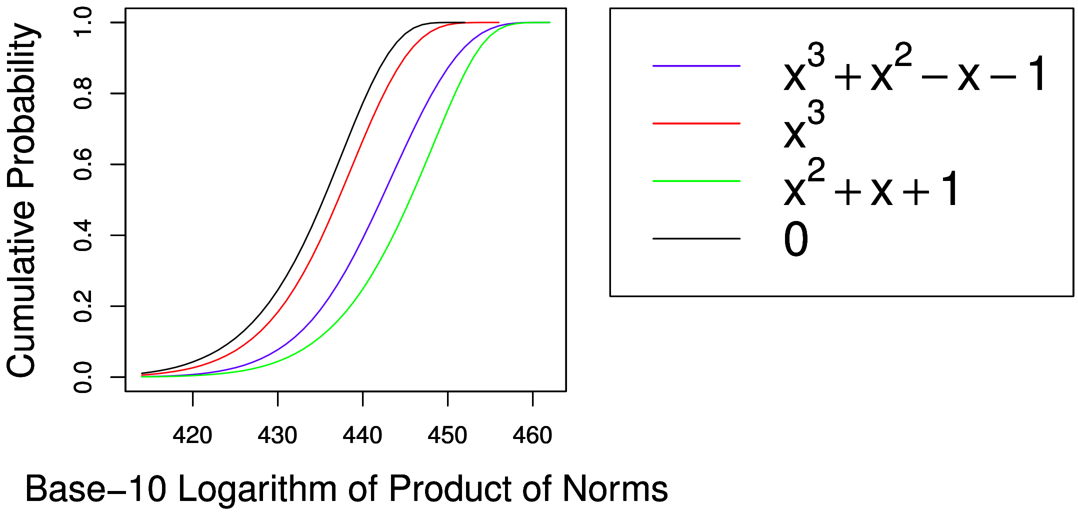

Figure 4.3: Plot of the cumulative probability that the base-10 logarithm of the BN product of the norms of a random element of the sieving space is at most some amount. Based on 500000 elements using $u = 6790547177159395210$.

Percentage of norms		10%	25%	50%	75%	90%	100%
$r(x)$	0	424.5	430.1	435.4	439.6	442.7	451.7
	x^3	426.5	431.9	437.1	441.5	444.9	455.1
	x^3+x^2-x-1	431.3	436.7	442.2	447.0	450.9	461.6
	x^2+x+1	434.3	440.0	445.5	449.9	453.0	462.0

Table 4.4: Percentiles for BN product of norms. The percentile represents the base-10 logarithm of the product of the norms below which the given percent of norm products fall (cf. Figure 4.3).

We see that the order of $r(x)$ choices, from best to worst, is 0, x^3, x^3+x^2+1, x^2+x+1. Although x^2+x+1 has lower degree than x^3, it has more terms and produces a polynomial f_2 with more terms and larger coefficients. We also see that for all four choices for $r(x)$, none of the selected elements of the sieving space have a norm product anywhere near as large as the theoretical upper bound of 6.284×10^{583} from Equation (3.12).

Based on this data, it would seem as though the number of terms and maximum size of the coefficients in f_2 has a much more significant relationship on norm sizes than does the degree of $r(x)$. To see how the number of terms and size of coefficients influence the norm, consider the norm as the determinant of a Sylvester matrix. If f_2 has very few coefficients, then the Sylvester matrix will be sparser and its determinant will likely be smaller. Similarly, if the coefficients of f_2 are small, then the determinant of the Sylvester matrix should be lower.

In Appendix A.1 we give for every polynomial $r(x)$ of degree ≤ 3 whose coefficients are all either 1, -1 or 0, both the number of terms and the coefficient with largest absolute value in the corresponding polynomial f_2. Note that these values are valid for any value of u. As such, we propose a simple method for selecting which $r(x)$ value to use for a specific DLP based on BN curves. First, determine which $r(x)$ choices give an irreducible polynomial f_1. Then choose a valid $r(x)$ that produces a polynomial f_2 with as few terms, and as small terms, as possible.

However, how much of an effect the choice of $r(x)$ actually has on the running time of the NFS is somewhat difficult to judge. Among all the percentiles listed in Table 4.4, the smallest was 424.5, while the largest was 462.0, less than 10% more than the smallest. However, we also see that 75% of the test elements for $r(x) = 0$ had norm less than the 25% percentile for $r(x) = x^2 + x + 1$, which seems to be a significant difference. It would be interesting to use Theorem 3.1 to estimate smoothness probabilities and determine the expected number of generated relations for each $r(x)$ choice. In Section 5.2, we will briefly look at how our data on the distribution of norms could potentially be used to speed up the NFS.

Another topic which merits further research is to look at the use of polynomials $r(x)$ with degree larger than three. If such a polynomial produced a sparse f_2 with small terms, than it could be a good choice for $r(x)$ despite its higher degree.

The maximum observed K_1 and K_2 norms for each of the four tested $r(x)$ values are give in Appendix A.3. As was the case in Section 4.2.1, all four $r(x)$ values had very similar maximum values for the K_1 norm.

4.2.3 Choice of u

For our final BN experiment, we look at how the choice of u affects the distribution of norms. By doing so, we hope to establish whether certain $r(x)$ values are always more likely to produce small norms, regardless of the choice of u.

Note that u has no effect on f_2, and hence on the K_2 norm. However, the polynomial $f_1(x) = x^{12} + r(x) - u$ is directly influenced by u. We expect that larger u values should produce somewhat larger norm values, since f_1 has a somewhat larger (in absolute value) constant term.

Since we are only concerned with values of u such that $p = P(u) = 36u^4 + 36u^3 + 24u^2 + 6u + 1$ is a 256-bit prime p, all the u values being considered will have approximately the same size. The smallest and largest u values that work are 6332666225848379031 and 7530851732716320751, respectively.

To look at the effect of u, we will look at distributions using the same $r(x)$ value but different u values. We do this for three different $r(x)$ values, namely $0, x^3$ and x^3+x^2+x-1. We chose to use 0 and x^3+x^2+x-1 since they seem to give lots of small and large norms, respectively. We searched for the smallest and largest u values for which each of these $r(x)$ choices gave an irreducible polynomial f_1. These were found to be

$$u_{\text{min}} = 6332666225848389958$$

and

$$u_{\text{max}} = 7530851732716310718$$

respectively. We also chose a third value of

$$u_{\text{mid}} = 6931758979282357578,$$

roughly half way between u_{min} and u_{max}, for which each of the above three $r(x)$ choices worked.

We calculated the norms for 500000 randomly chosen elements of the sieving space, for each choice of $r(x)$ and each choice of u. We list in Table 4.5 some percentiles for $r(x) = 0$. Similar data for $r(x) = x^3$ and $x^3 + x^2 + x - 1$ is listed in Appendix A.2.

Percentage of norms	10%	25%	50%	75%	90%	100%
u_{min}	424.2	429.8	435.0	439.3	442.3	451.0
u_{mid}	424.6	430.2	435.5	439.7	442.8	451.2
u_{max}	424.9	430.6	435.8	440.1	443.2	452.0

Table 4.5: Percentiles for BN product of norms for $r(x) = 0$. The percentile represents the base-10 logarithm of the product of the norms below which the given percent of norm products fall. Based on 500000 elements.

As expected, higher u values produce somewhat larger norms. However, the differences appear to be quite insignificant, even when comparing the largest and smallest possible u values.

Benger *et al.* noted in their paper [9] that for MNT curves, the size of the prime p affects only the placement of the distribution, not the shape. Based on Table 4.5, we see that a similar pattern seems to apply for BN curves. For each of the three u values tested, the differences between percentiles are essentially the same. For example, the values of the 50% percentiles minus the 25% percentiles are about 5.2, 5.3 and 5.2. Also, each of the listed percentiles for u_{mid} (except the 100% percentile) is about 0.4 higher than the same percentile for u_{min}. A similar relationship holds for u_{max} and u_{mid}.

The same patterns also apply for the other two choices of $r(x)$. Interestingly, the same difference of about 0.4 between percentiles for u_{mid} and u_{min} holds for both of these other $r(x)$.

As a side note, none of the tested u and $r(x)$ values produced a norm product anywhere near as large as the theoretical upper bound in Equation (3.12). The maximum observed K_1 and K_2 norms for each of the four tested $r(x)$ values are give in Appendix A.3. Once again, the choice of $r(x)$ had very little effect on the maximum values for the K_1 norm, although larger u values produced larger maximums.

We conclude from this experiment that if a given choice of $r(x)$ gives lots of small norms for one value of u, then it will likely gives lots of small norms for all u values. This is fortunate, as it means that testing to determine an optimal $r(x)$ value must only be done for a single u value.

Chapter 5

Analysis of Results

Having developed a method for collecting experimental data on the distribution of norm values for BN curves, we now aim to model the distribution of the sizes of the norms. We then discuss how developing such a model could potentially allow us to develop a more efficient NFS implementation by only attempting to factorize elements having norms below some calculated upper bound during the sieving stage.

5.1 Modelling the Distribution of Norms

Our goal in this section is to attempt to model the distribution of the norms for BN curves. We will look at each of the two norms individually.

Benger *et al.* [9] worked with the distribution of K_1 norms in the MNT setting. Let Y be the fraction of elements in the sieving space with K_1 norm bounded above by X. They attempted to apply transformations to X in order to find a linear relationship between Y and X. Working under the assumption that the distribution of norms is of the form

$$Y^{\lambda} = aX + b, \tag{5.1}$$

for some constant λ, they used the Box-Cox method [13] to find the optimal value for λ. The idea behind the Box-Cox method is to look at a series of transformations of the form $T(Y) = (Y^{\lambda} - 1)/\lambda$, and to find the value of λ for which the relationship between $T(Y)$ and X is as close to linear as possible. A brief overview of the Box-Cox method can be found in the NIST Engineering Statistics Handbook [1, Section 1.3.3.5 – Box-Cox Linearity Plot].

As mentioned in Section 4.2.3, the size of u and the corresponding prime $p = P(u)$ appears to have no influence on the shape of the norm distribution. As such, Benger *et al.* worked with a 50-bit prime p. They determined via the Box-Cox method that the fraction Y of elements with f_1 norms bounded above by X can be approximated by

$$Y^6 = aX + b,$$

where $a = 2.171 \times 10^{-106}$ and $b = -1.868 \times 10^{-5}$. Note that the optimal value for $\lambda = 6$ is equal to the degree of the polynomial f_1 used in the MNT setting.

We are interested in applying a similar analysis to BN curves. We also start by assuming that the distribution of the norms is of the form given above in Equation (5.1). Further, we make the hypothesis that the optimal value of λ for the K_i norm will be equal to $\deg f_i$. To support our hypothesis, we note that given an element $\alpha = \sum_{j=0}^{d_i-1} a_j x^j + (f_i(x)) \in K_i$, the norm function $N_{K_i}(\alpha) = \mathrm{Res}_x(\sum_{j=0}^{d_i-1} a_j x^j, f_i(x))$ can be viewed as a polynomial function of the d_i coefficients $a_0, \ldots, a_{d_i}$ of α. In fact, it follows from the definition given in Section 2.3.1 of the resultant as the determinant of a Sylvester matrix, that the norm is a homogeneous polynomial of degree twelve. Thus the values output by the norm function are simply sums of monomials, where each monomial is a product of twelve randomly chosen integers between 1 and the sieving bound S. As noted by Benger *et al.*, we should expect a significant clumping effect around the middle section of the range of possible norm values.

To generate data for use with the Box-Cox method, we randomly selected

$$u = 6928097775065274527$$

with $r(x) = x^3 - 1$, and with other parameters chosen as in Section 4.2. We picked 500000 random elements of the sieving space, calculated their norms and recorded them separately. The highest observed norms for K_1 and K_2 were

$$M_1 \approx 8.944 \times 10^{250}$$

and

$$M_2 \approx 1.321 \times 10^{205}$$

respectively.

Due to the much larger numbers involved in this setting, we will look at the fraction Y of elements with norms bounded above by X times the maximum observed norm M_i, and attempt to determine the optimal value of λ such that Y^λ is linear in X. Since we are simply multiplying our X values by $1/M$, we should get the same optimal λ value. A plot of the cumulative probability that a randomly chosen element α of the sieving space will have $N_{K_1}(\alpha)$ at most some fraction of M_i is given in Figure (5.1). A similar plot for K_2 in given in Figure 5.2. Both plots were constructed in the same way as described in Section 4.1, with the exception that we now choose to plot the data points themselves, rather than the line segments between points. We also do not include a point at the origin, causing the range of cumulative probability values to be much smaller, especially for the K_2 norms.

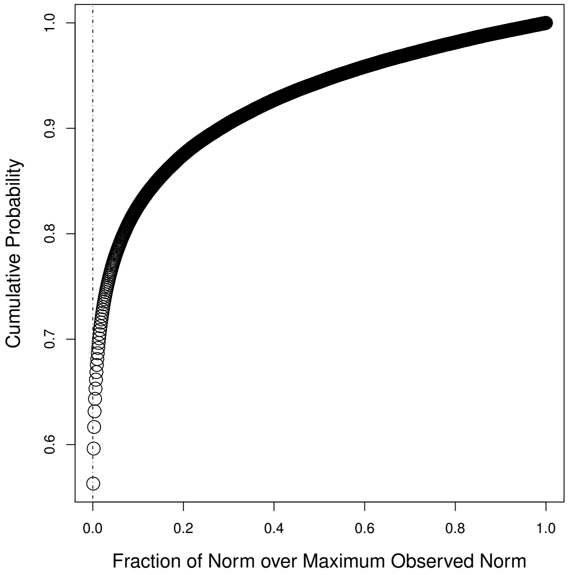

Figure 5.1: Plot of the cumulative probability Y that the BN K_1 norm of a random element of the sieving space is at most some fraction X of the maximum observed norm $M_1 \approx 8.944 \times 10^{250}$. Based on 500000 norm values, for $u = 69280977750652745 27$.

Using an implementation of the Box-Cox method provided in the MASS package [51] for the statistical computing language R [44], we tested λ values from 1 to 20 for the K_1 norm. The best value was clearly $\lambda = 12$, which, as predicted, corresponds to the degree of f_1. Detailed output from the package is available in Appendix A.4.

We then used the R function "lm" to find the best linear model between X and Y^{12}, obtaining the equation

$$Y^{12} = 0.001557 + 0.9980X.$$

In Figure 5.3, we add the transformed data points to the plot of the K_1 norm distribution from Figure 5.1. We checked the fit of the model using the R function "summary.lm". Its

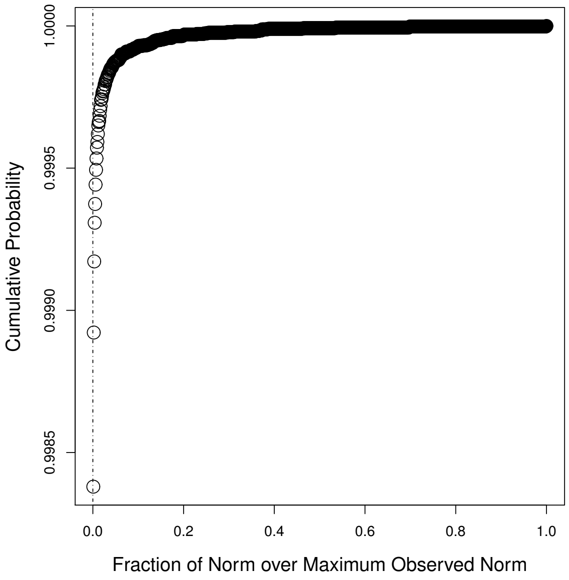

Figure 5.2: Plot of the cumulative probability Y that the BN K_2 norm of a random element of the sieving space is at most some fraction X of the maximum observed norm $M_2 \approx 1.321 \times 10^{205}$ (cf. Figure 5.1). Note that the range of cumulative probability values is much smaller than in Figure 5.1.

output is given in Appendix A.4. For more information on the meaning of the values output by "summary.lm", consult the R reference manual, particularly the page on "summary.lm".

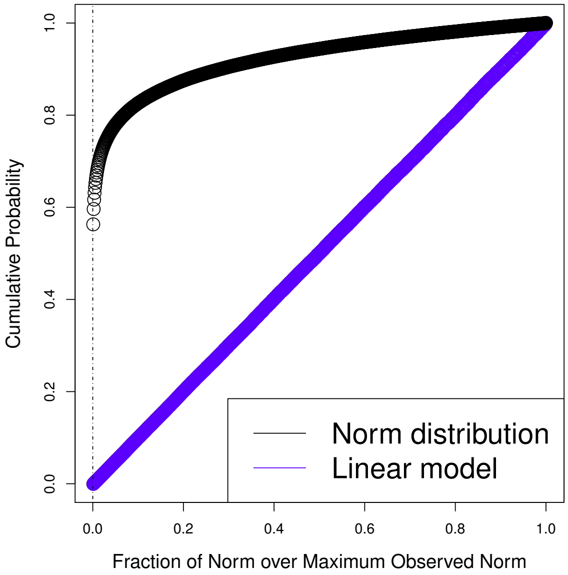

Figure 5.3: Plot for the BN K_1 norm, of the cumulative probability Y and the transformed cumulative probability Y^{12}, versus X (cf. Figure 5.1).

When working with the degree 48 polynomial f_2, the Box-Cox method gave less definitive results. Note that the X-values of our data points are based on how we choose to divide up our norms into bins. The bins used to generate K_1 norm data were the same as in Section 4.1, namely

$$\{((i-1)/1000, i/1000] : 1 \leq i \leq 1000, i \in \mathbb{Z}\}.$$

The X-values are simply the upper bounds on each bin. This choice of bins work well to model the K_1 norm. However, for the K_2 norm the cumulative probability grows too quickly and we end up with too many data points with very high Y-values. Running the Box-Cox method with this choice of bins gives better results for high λ values. We plot the norm distribution data, as well as the transformed data for λ values of 20, 50 and 100 in Figure 5.4. The transformed data is not very smooth, and includes large numbers of points with the same Y value. This can be explained by observing that very few of the norm values calculated for the experiment had values anywhere near as large as the maximum M_2. Thus there are large jumps among the highest norm values, leading to lots of empty buckets and lots of points with the same Y value. This is confirmed in Figure 5.5 in which we plot data points for each element tested in our experiments, instead of using one point for each bucket.

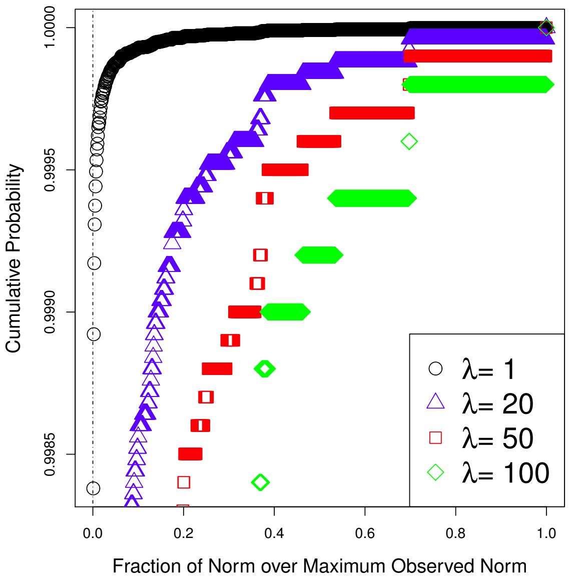

Figure 5.4: Plot for the BN K_2 norm of Y^λ versus X, for various λ values (cf. Figure 5.2).

We thus attempt to come up with a different way of choosing buckets for the K_2 norm. We would like the X values input to the Box-Cox method to more accurately match the X-values in Figure 5.5. As a first attempt, we create one bucket for each element. However, this ends up giving a very high number of points with very low X-values. As one might expect from looking at Figure 5.5, the Box-Cox method returns $\lambda = 1$ in this case.

For the Box-Cox method to return a reasonable λ value, we need most of our X-values to be around the curved part of the distribution. To do so, we start by sorting the 500000 experimentally determined points based on their norm values, in increasing order.

We then attempt to select a subset of these points to accurately capture the shape of the curve. To do so, we would like to avoid choosing too many points with very similar X or Y-values. We chose to take the first point, and then to recursively take the next lowest point whose X-value was bigger than the previous point by at least 0.05, or whose Y-value was bigger than the previous point by at least 0.000001. We obtained 1315 points through this process, which are plotted in Figure 5.6.

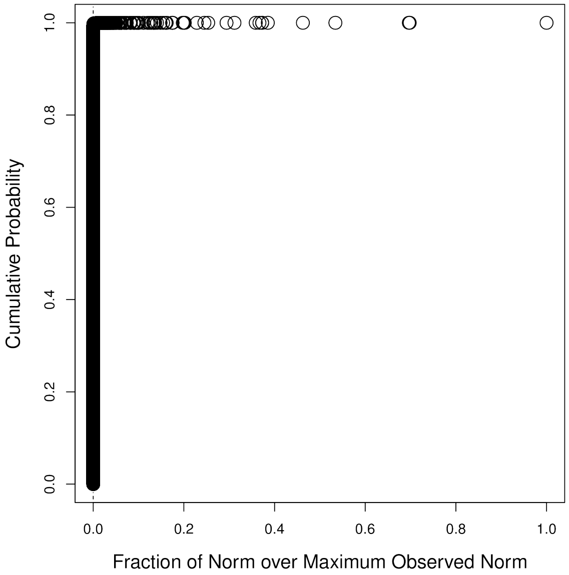

Figure 5.5: Plot of Y versus X for the BN K_2 norm, showing one point for each of the 500000 tested elements of the sieving space (cf. Figure 5.2).

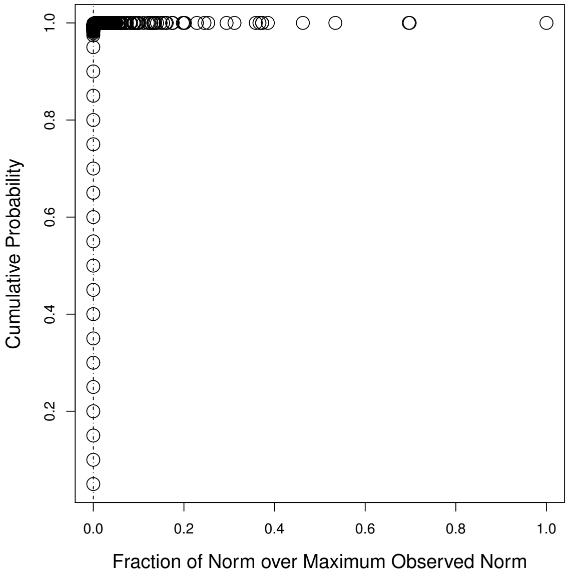

Figure 5.6: Plot of Y versus X for the BN K_2 norm, using only points that are not too close together (cf. Figure 5.5). These were the points used as input for the Box-Cox method.

Using these points as input to the Box-Cox method, we tested λ values between 30 and 70 and found that $\lambda = 50$ is optimal, although values close to 50 were also indicated to be likely. Detailed output from the Box-Cox method is available in Appendix A.4.

We also repeated the same procedure three more times for the same value of u and $r(x)$. The same optimal value for K_1 of $\lambda = 12$ was obtained each time. However, for K_2 five different optimal values were obtained: 40, 25, 47, 29 and 35. Based on qualitative observations of plots similar to Figure 5.6 that were generated for each data set, it would appear as though larger λ values corresponded to sets with a relatively higher number of norm values close to the maximum observed value.

The set of points that we input to the Box-Cox method still seems to be in need of some refinement. The results returned by the Box-Cox are of course dependent on the input points, and choosing a set of inputs that correctly captures the behaviour of the norm distribution is difficult. Out of the 500000 norms calculated for the experiments, our chosen set of input points included the highest 300 norms. This likely causes λ to be too heavily dependent on the distribution of the these highest norms.

To consistently get the same λ value, we likely need to find a way of selecting a good set of input points that does not include too many of the very large norms. However, a simpler approach such as just calculating more norm values could also help reduce experimental error and produce a more accurate answer.

5.2 Effect on the Number Field Sieve

We now turn to an analysis of the effect of the distribution of the norms on the running time of the NFS. Ignoring the difficulties discussed in Section 5.1 that arise when attempting to model the K_2 norm, we will assume that we have a model which accurately predicts the distribution of norm values for a given set of parameters. Suppose that with this model, we can accurately estimate, for any constant C, the approximate number of elements in the entire sieving space for which the absolute value of the norm product is at most C. We briefly discuss here how this information could potentially be used to come up with a better NFS implementation.

The key idea behind improving the NFS running time is that during the sieving step, we can choose a constant A and skip over any element α of the sieving space for which the absolute value of the product of the norms of α is larger than A. Since these elements have very large norm, we expect that they will be unlikely to be B-smooth. As such, we can avoid testing them for smoothness and immediately pick another element of the sieving space. By only performing smoothness tests on elements that are more likely to be smooth, we hope to reduce the total number of smoothness tests required in the sieving stage. Since checking elements for smoothness constitutes most of the work in the sieving stage, we thus hope to reduce the stage's overall running time.

Our model of the norm distribution will allow us to predict the approximate number of elements α that will be skipped. By choosing A sufficiently large, we can choose our parameters so that we still expect to be able to find sufficiently many relations among the elements that we test for smoothness.

To come up with an accurate estimate for A, we must once again rely on Theorem (3.1) in order to relate the size of norms to the probability of smoothness. Although it would be preferable to have experimental data on BN curves to back up the theorem, this would require attempting to factorize a very large number of very large norms. Performing such an experiment is beyond the scope of this thesis.

Chapter 6

Theoretical Upper Bound

In this chapter we will attempt to find better theoretical upper bounds for BN curves on the absolute value of the product of the norms.

In Section 3.7.2, we proved an upper bound of

$$|N_{K_1}(\alpha)N_{K_2}(\alpha)| \leq \left((n_1+t)!\, S^{n_1}\,(u+1)^t\right)\left((4n_1+t)!\, S^{4n_1}\cdot(180(2+\deg r)^4)^t\right)$$

in Equation (3.12). Using the parameters of $n_1 = \deg f_1 = 12$, $t = 11$ and $S = 4387$ that we calculated in Section 3.7.2, we get an upper bound of

$$|N_{K_1}(\alpha)N_{K_2}(\alpha)| \leq 180^{11}\cdot 23!\cdot 59!\cdot 4387^{60}(u+1)^{11}(2+\deg r)^{44}. \tag{6.1}$$

In Chapter 4, we calculated millions of norm values using these parameters, yet failed to find any elements of the sieving space whose norm product had an absolute value anywhere near this large. For each experiment in Chapter 4, we recorded the u and $r(x)$ values used, as well as the maximum absolute values of the K_1 norm, the K_2 norm and their product. These maximum observed values are given in Appendix A.3. Among all the experiments, the largest K_1 norm was about 2.239×10^{251} and the largest K_2 norm was about 2.232×10^{211}. However, even evaluating Equation (6.1) using lower bounds on u and $\deg r$ of $u = 2^{62}$ and $\deg r = 0$, we still get an upper bound of 2.756×10^{564}. Based on experimental evidence, this upper bound does not appear to be very tight.

Our goal in this chapter will be to prove upper bounds that are closer to these maximum observed values. In particular, we will compare the theoretical and experimental bounds for $u = u_1 = 6917529027641089837$ and $r(x) = x^3$.

We start by looking at the norms individually, comparing theoretical upper bounds for the K_i to the maximum observed K_i norm absolute values. By doing so, we hope to determine approximately how much each of the two bounds is in need of improvement. Breaking up Equation 3.12 into separate bounds for each norm, we get that

$$\begin{aligned}|N_{K_1}(\alpha)| &\leq (n_1+t)!\, S^{n_1}\,(u+1)^t \\ &= 23! \cdot 4387^{12}(u+1)^{11}\end{aligned}$$

and

$$\begin{aligned}|N_{K_2}(\alpha)| &\leq (4n_1+t)!\, S^{4n_1} \cdot (180(2+\deg r)^4)^t \\ &= 59! \cdot 4387^{48} \cdot 180^{11} \cdot (2+\deg r)^{44}.\end{aligned}$$

For $u = 6917529027641089837$ and $r(x) = x^3$, the maximum observed K_1 and K_2 norms were 8.796×10^{250} and 1.328×10^{204} respectively. The theoretical bounds given above evaluate to about 2.280×10^{273} and 3.379×10^{310} respectively. Based on this, it would appear as though both theoretical bounds can be tightened, although the bound on K_1 is much tighter.

To strengthen our theoretical bounds, we start by proving the following variant of Lemma 3.2. Given a square matrix $M = [m_{ij}]_{1\leq i,j\leq m}$, we let $M_{i,j}$ denote the submatrix of M obtained by removing the i-th row and j-th column.

Lemma 6.1. *Let A be an $m \times m$ matrix such that the sum of the absolute values of the entries in the i-th row of A is bounded above by B_i, for $i = 1, \ldots, m$. Then $|\det A| \leq \prod_{i=1}^m B_i$.*

Proof. If A is the 1×1 matrix $[a]$ then we can take $B_1 = |a|$ to get $|\det A| = |a| \leq B_1$. Suppose the theorem holds for all $(m-1) \times (m-1)$ matrices. Let $A = [a_{ij}]_{1\leq i,j\leq m}$ be an $m \times m$ matrix such that the sum of the absolute values of the entries in the i-th row of A is bounded above by B_i, for $i = 1, \ldots, m$. From the Laplace expansion for the determinant, we get that

$$
\begin{aligned}
|\det A| &= \left| \sum_{j=1}^{m} a_{1j}(-1)^{1+j} \det A_{1,j} \right| \\
&\leq \sum_{j=1}^{m} |a_{1j} \det A_{1,j}| \\
&\leq \sum_{j=1}^{m} \left| a_{1j} \prod_{i=2}^{m} B_i \right| \\
&= \left(\prod_{i=2}^{m} B_i \right) \sum_{j=1}^{m} |a_{1j}| \\
&\leq \prod_{i=1}^{m} B_i
\end{aligned}
$$

where the second inequality follows from induction since B_{i+1} is an upper bound on the absolute values of the entries in the i-th row of $A_{1,j}$, for $i = 1, \ldots, m-1$ and $j = 1, \ldots, m$. □

As was done in Chapter 3, we apply this theorem to the Sylvester matrix A_i of f_i and $\sum_{j=0}^{t} a_j x^j$ in order to get an upper bound on the norm. In each of the first t rows of A_i, the only non-zero entries are the coefficients of f_i. So for f_1, the sum of the absolute values is bounded above by $u + \deg r + 1$. For f_2, getting a tight bound is harder. However, our bound of

$$180(2 + \deg r)^4$$

from Equation (3.11) is actually an upper bound on the sum of the absolute values of the coefficients of f_2.

For the last $\deg f_i = n_i$ rows of A, all entries have absolute values bounded above by $S = 4387$ and each row has at most $t + 1 = 12$ entries.

We thus get new theoretical upper bounds on the K_1 and K_2 norms of

$$
\begin{aligned}
|N_{K_1}(\alpha)| &\leq (u + \deg r + 1)^t \cdot (S(t+1))^{n_1} \\
&= (u + \deg r + 1)^{11} \cdot (4387 \cdot 12)^{12}
\end{aligned}
$$

and

$$
\begin{aligned}
|N_{K_2}(\alpha)| &\leq ((n_2 + 1) \cdot 180(2 + \deg r)^4)^t \cdot (S(t+1))^{n_2} \\
&= (180(2 + \deg r)^4)^{11} \cdot (4387 \cdot 12)^{48}.
\end{aligned}
$$

For $u = 6917529027641089837$ and $r(x) = x^3$, these new theoretical bounds on the K_1 and K_2 norms evaluate to about 7.864×10^{263} and 1.540×10^{282} respectively. This a slight improvement, but both bounds can likely be further lowered.

We next observe that we can combine Lemmas 3.2 and 6.1 to get a slightly tighter bound.

Lemma 6.2. *Let A be an $m \times m$ matrix. For $i = 1, \dots, m$, let B_i be an upper bound on the sum of the $(m+1-i)$ entries in the i-th row of A with largest absolute value. Then $|\det A| \leq \prod_{i=1}^{m} B_i$.*

Proof. If A is the 1×1 matrix $[a]$ then we can take $B_1 = |a|$ to get $|\det A| = |a| \leq B_1$. Suppose the theorem holds for all $(m-1) \times (m-1)$ matrices. Let $A = [a_{ij}]_{1 \leq i,j \leq m}$ be an $m \times m$ matrix. For $i = 1, \dots, m$, let B_i be an upper bound on the sum of the $(m+1-i)$ entries in the i-th row of A with largest absolute value. From the Laplace expansion for the determinant, we get that

$$\begin{aligned}
|\det A| &= \left| \sum_{j=1}^{m} a_{1j} (-1)^{1+j} \det A_{1,j} \right| \\
&\leq \sum_{j=1}^{m} |a_{1j} \det A_{1,j}| \\
&\leq \sum_{j=1}^{m} \left| a_{1j} \prod_{i=2}^{m} B_i \right| \\
&= \left(\prod_{i=2}^{m} B_i \right) \sum_{j=1}^{m} |a_{1j}| \\
&\leq \prod_{i=1}^{m} B_i.
\end{aligned}$$

The second inequality follows from induction since $A_{1,j}$ is an $(m-1) \times (m-1)$ matrix, and B_{i+1} is an upper bound on the sum of the $((m-1)+1-i)$ entries in the i-th row of $A_{1,j}$ with largest absolute value, for $i = 1, \dots, m-1$ and $j = 1, \dots, m$. □

For $i = 1, \dots, t$, the sum of the $(n_1 + t + 1 - i)$ entries of the i-th row of A_1 with largest absolute value can be bounded above by $u + \deg r + 1$. A slightly stronger bound is possible for the last few of these rows, but it isn't very helpful. For each of the first t rows of A_2, we will use the same bound of $180(2+\deg r)^4$ as before. For the last n_i rows of A_i, we can bound the i-th row by $\min(n_i+t+1-i, t+1)S$, since each row has only $t+1$ non-zero entries.

Our new theoretical upper bounds are given by

$$\begin{aligned}|N_{K_1}(\alpha)| &\leq (u+\deg r+1)^t \cdot n_1! \cdot S^{n_1}\\ &= (u+\deg r+1)^{11} \cdot 12! \cdot 4387^{12}\end{aligned}$$

and

$$\begin{aligned}|N_{K_2}(\alpha)| &\leq (180(2+\deg r)^4)^t \cdot n_1^{n_2-n_1} \cdot n_1! \cdot S^{n_2}\\ &= (180(2+\deg r)^4)^{11} \cdot 12^{36} \cdot 12! \cdot 4387^{48}.\end{aligned}$$

For $u = 6917529027641089837$ and $r(x) = x^3$, these theoretical bounds on the K_1 and K_2 norms evaluate to about 4.225×10^{259} and 8.272×10^{277} respectively. These are again slight improvements over the previous bounds. Our theoretical bound on the K_1 norm is now getting close the experimental bound of 8.796×10^{250}. However, our bound for K_2 is still much larger than the largest observed norm value of 1.328×10^{204}.

We mentioned in Section 4.2.1 that the choice of $r(x)$ seems to have very little influence on upper bounds for the K_1 norm. In fact, the choice of $r(x)$ only influences the upper bound by slightly altering the upper bounds that we can use for the first t rows of A_1. For any choice of $r(x)$ of degree ≤ 3, the sum of the absolute values of the coefficients of f_1 will be between $u-\deg r$ and $u+2+\deg r$. This is a very slight difference compared to the size of u.

One of the problems with our bound on the K_2 norm is that our bound on the sum of the $(m+1-i)$ entries in the i-th row of A with largest absolute value is not very tight. To get a tighter bound, we look at the actual values of the coefficients of $f_2 = P(x^{12}+r(x))$. With our choice of $r(x) = x^3$, we get

$$\begin{aligned}f_2 =& 36x^{48} + 144x^{39} + 36x^{36} + 216x^{30} + 108x^{27} + 24x^{24} + 144x^{21} + 108x^{18}\\ &+ 48x^{15} + 42x^{12} + 36x^9 + 24x^6 + 6x^3 + 1.\end{aligned}$$

For the first $t = 11$ rows of the Sylvester matrix, we get bounds of 973, 972, 966, 942, 918, 882, 846, 810, 768, 720, 612, 504, 360 and 216. These are all much lower than our previous bounds of $180(2+\deg r)^4 = 112500$.

With these new row bounds, our theoretical bound for K_2 becomes

$$\begin{aligned} |N_{K_2}(\alpha)| \leq \quad & 973 \cdot 972 \cdot 966 \cdot 942 \cdot 918 \cdot 882 \cdot 846 \cdot 810 \cdot 768 \cdot 720 \cdot 612 \cdot 504 \cdot 360 \cdot 216 \cdot \\ & 12^{36} \cdot 12! \cdot 4387^{48} \\ \approx \quad & 1.434 \times 10^{262}. \end{aligned}$$

Unfortunately this was the lowest theoretical bound that we were able to prove. It seems likely that further work could be done to improve this bound.

Chapter 7

Conclusion

This thesis investigated issues relating to the implementation of the number field sieve for specific instances of discrete logarithm problems arising from Barreto-Naehrig curves. We found some bounds on the product of the norms, and used those bounds to find running time estimates for the NFS. We then found specific parameter values for the sieving bound, sieving degree and smoothness bound that optimize the running time of the NFS in the 80-bit MNT and 128-bit BN security settings, using the choice of polynomials suggested by Joux *et al.* [28]. This was also done for BN curves using the choice of polynomials suggested by Joux and Pierrot [30]. We showed that Joux-Pierrot polynomials allow for a faster implementation of the NFS.

Using Joux-Pierrot polynomials with BN curves, we collected experimental data on the distribution of the sizes of the norms in order to better estimate the number of relations that we can expect to find in the sieving stage. Based on this data, we made some suggestions on how to pick a good choice of Joux-Pierrot polynomials for specific DLPs.

We then used the Box-Cox method to find a good model for the distribution of the sizes of the norms. Using this information, it may be possible to speed up the NFS by taking advantage of the large number of elements with norm much smaller than the theoretical upper bound. Finally, having observed that our theoretical bound on the product of the norms did not appear to be very tight, we proved some bounds on determinants that enabled us to find better bounds for the norm product.

Chapter 8

Future Work

It seems likely that the experimental data generated in Chapter 4 and the norm distribution model developed in Chapter 5 can be used to help speed up the NFS. One possible approach is outlined in Section 5.2, but more work needs to be done to understand exactly how this information impacts the running time of the NFS.

It could also be useful to test more of the assumptions on which our running time analysis is based. We used Theorem 3.1 to approximate the probability that a norm of a given size will be used. We also assumed that for an element α in the sieving space, the probability of $N_{K_1}(\alpha)$ and $N_{K_2}(\alpha)$ being B-smooth were independent. The correctness of both of these assumptions could be analyzed.

Another area in which future work may be beneficial is in obtaining even lower theoretical bounds on the absolute value of the norm for Barreto-Naehrig curves.

Finally, the use of polynomials $r(x)$ of degree greater than three could be viable to generate good Joux-Pierrot polynomials. However, we would need to look for choices of $r(x)$ for which f_2 has small coefficients and relatively few terms.

References

[1] NIST/SEMATECH e-Handbook of Statistical Methods. http://www.itl.nist.gov/div898/handbook/. Accessed: 2015-04-18.

[2] A. Atkin and F. Morain. Elliptic curves and primality proving. *Mathematics of Computation*, 61:29–68, 1993.

[3] R. Balasubramanian and N. Koblitz. The improbability that an elliptic curve has subexponential discrete log problem under the Menezes-Okamoto-Vanstone algorithm. *Journal of Cryptology*, 11(2):141–145, 1998.

[4] R. Barbulescu, P. Gaudry, A. Guillevic, and F. Morain. Improvements to the number field sieve for non-prime finite fields. https://hal.inria.fr/hal-01052449, November 2014. Accessed: 2014-12-01.

[5] R. Barbulescu, P. Gaudry, A. Joux, and E. Thomé. A heuristic quasi-polynomial algorithm for discrete logarithm in finite fields of small characteristic. In *Advances in Cryptology – EUROCRYPT 2014*, volume 8441 of *Lecture Notes in Computer Science*, pages 1–16. Springer-Verlag, 2014.

[6] R. Barbulescu and C. Pierrot. The multiple number field sieve for medium- and high-characteristic finite fields. *LMS Journal of Computation and Mathematics*, 17(A):230–246, 2014.

[7] E. Barker, W. Barker, W. Burr, W. Polk, and M. Smid. Recommendation for Key Management - Part 1: General (Revision 3). http://csrc.nist.gov/publications/nistpubs/800-57/sp800-57_part1_rev3_general.pdf, July 2012. Accessed: 2014-12-01.

[8] P. Barreto and M. Naehrig. Pairing-friendly elliptic curves of prime order. In *Selected Areas in Cryptography – SAC 2005*, volume 3897 of *Lecture Notes in Computer Science*, pages 319–331. Springer-Verlag, 2006.

[9] N. Benger, M. Charlemagne, and K. Chen. A note on the behaviour of the NFS in the medium prime case: Smoothness of norms. Personal communication, March 2014. An earlier version can be found as eprint 2013/147, http://eprint.iacr.org/2013/147/.

[10] D. Boneh and M. Franklin. Identity-based encryption from the Weil pairing. In *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–229. Springer-Verlag, 2001.

[11] Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the Weil pairing. *Journal of Cryptology*, 17(4):297–319, 2004.

[12] Wieb Bosma, John Cannon, and Catherine Playoust. The Magma algebra system. I. The user language. *Journal of Symbolic Computation*, 24:235–265, 1997. http://magma.maths.usyd.edu.au/magma/.

[13] G. E. P. Box and D. R. Cox. An analysis of transformations. *Journal of the Royal Statistical Society. Series B (Methodological)*, 26(2):211–252, 1964.

[14] E. R. Canfield, P. Erdős, and C. Pomerance. On a problem of Oppenheim concerning "factorisatio numerorum". *J. Number Theory*, 17(1):1–28, 1983.

[15] M. Charlemagne. C library for use with distribution of norms. Personal communication, March 2014.

[16] H. Cohen. *A Course in Computational Algebraic Number Theory*, volume 138 of *Graduate Texts in Mathematics*. Springer, 1993.

[17] D. Coppersmith, A. M. Odlzyko, and R. Schroeppel. Discrete logarithms in GF(p). *Algorithmica*, 1(1-4):1–15, 1986.

[18] A. J. Devegili, M. Scott, and R. Dahab. Implementing cryptographic pairings over Barreto-Naehrig curves. In *Pairing-Based Cryptography – Pairing 2007*, volume 4575 of *Lecture Notes in Computer Science*, pages 197–207. Springer Berlin Heidelberg, 2007.

[19] W. Diffie and M. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

[20] T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472, 1985.

[21] L. Fousse, G. Hanrot, V. Lefèvre, P. Pélissier, and P. Zimmermann. MPFR: A Multiple-precision Binary Floating-point Library with Correct Rounding. *ACM Transactions on Mathematical Software*, 33(2), 2007. Version 3.1.2.

[22] G. Frey and H. Rück. A remark concerning m-divisibility and the discrete logarithm in the divisor class group of curves. *Mathematics of Computation*, 62(206):865–874, 1994.

[23] D. Gordon. Discrete logarithms in GF(p) using the number field sieve. *SIAM Journal on Discrete Mathematics*, 6(1):124–138, 1993.

[24] T. Granlund and the GMP development team. *GNU MP: The GNU Multiple Precision Arithmetic Library*, 6.0.0 edition, 2014. http://gmplib.org/.

[25] T. Granlund and the MPIR development team. *MPIR: Multiple Precision Integers and Rationals*, 2.6.0 edition, 2012. http://mpir.org/.

[26] W. Hart, F. Johansson, and S. Pancratz. *FLINT: Fast Library for Number Theory*, 2.4.3 edition, 2014. http://flintlib.org.

[27] A. Joux. A one round protocol for tripartite Diffie-Hellman. *Journal of Cryptology*, 17(4):263–276, 2000.

[28] A. Joux, R. Lercier, N. Smart, and F. Vercauteren. The number field sieve in the medium prime case. In *CRYPTO 2006*, volume 4117 of *Lecture Notes in Computer Science*, pages 326–344. Springer-Verlag, 2006.

[29] A. Joux, A. Odlyzko, and C. Pierrot. The past, evolving present and future of the discrete logarithm. In *Open Problems in Mathematics and Computational Science*, pages 5–36. Springer International Publishing, 2014.

[30] A. Joux and C. Pierrot. The special number field sieve in $\mathbb{F}_{p^n}$ – application to pairing-friendly constructions. In *Pairing-Based Cryptography – Pairing 2013*, volume 8365 of *Lecture Notes in Computer Science*, pages 45–61. Springer-Verlag, 2014.

[31] A. K. Lenstra and H. W. Lenstra Jr. *The Development of the Number Field Sieve*, volume 1554 of *Lecture Notes in Mathematics*. Springer, 1993.

[32] R. Lidl and H. Niederreiter. *Finite Fields*, volume 20 of *Encyclopedia of Mathematics and its Applications*. Addison-Wesley, 1983.

[33] A. Menezes. An introduction to pairing-based cryptography. In *Recent Trends in Cryptography*, volume 477 of *Contemporary Mathematics*, pages 47–65. AMS, 2009.

[34] A. J. Menezes, T. Okamoto, and S. A. Vanstone. Reducing elliptic curve logarithms to logarithms in a finite field. *IEEE Transactions on Information Theory*, 39(5):1639–1646, 1993.

[35] V. Miller. The Weil pairing, and its efficient calculation. *Journal of Cryptology*, 17(4):235–261, 2004.

[36] A. Miyaji, M. Nakabayashi, and S. Takano. New explicit conditions of elliptic curve traces for FR-reduction. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Science*, E84-A(5):1234–1243, 2001.

[37] F. Morain. Building cyclic elliptic curves modulo large primes. In *Advances in Cryptology – EUROCRYPT '91*, volume 547 of *Lecture Notes in Computer Science*, pages 328–336. Springer-Verlag, 1991.

[38] A. Odlyzko. Discrete logarithms: The past and the future. *Designs, Codes and Cryptography*, 19(2-3):129–145, 2000.

[39] A. M. Odlyzko. Discrete logarithms in finite fields and their cryptographic significance. In *Advances in Cryptology*, volume 209 of *Lecture Notes in Computer Science*, pages 224–314. Springer-Verlag, 1985.

[40] G. C. C. F. Pereira, M. A. Simplício Jr, M. Naehrig, and P. S. L. M. Barreto. A family of implementation-friendly BN elliptic curves. *Journal of Systems and Software*, 84(8):1319–1326, 2011.

[41] C. Pierrot. The multiple number field sieve with conjugation and generalized Joux-Lercier methods. In *Advances in Cryptology – EUROCRYPT 2015*, volume 9056 of *Lecture Notes in Computer Science*, pages 156–170. Springer-Verlag, 2015.

[42] J. Pollard. Monte Carlo methods for index computation mod p. *Mathematics of Computation*, 32(143):918–924, 1978.

[43] C. Pomerance. Elementary thoughts on discrete logarithms. In *Algorithmic Number Theory: Lattices, Number Fields, Curves and Cryptography*, volume 44 of *MSRI Publications*, pages 385–396. Cambridge University Press, 2008.

[44] R Core Team. *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria, 2014. http://www.R-project.org/.

[45] O. Schirokauer. Discrete logarithms and local units. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 345(1676):409–423, 1993.

[46] O. Schirokauer. Using number fields to compute logarithms in finite fields. *Mathematics of Computation*, 69(231):1267–1283, 2000.

[47] O. Schirokauer. The impact of the number field sieve on the discrete logarithm problem in finite fields. In *Algorithmic Number Theory: Lattices, Number Fields, Curves and Cryptography*, volume 44 of *MSRI Publications*, pages 397–420. Cambridge University Press, 2008.

[48] A. Shamir. Identity-based cryptosystems and signature schemes. In *Advances in Cryptology – Proceedings of CRYPTO 84*, volume 196 of *Lecture Notes in Computer Science*, pages 47–53. Springer-Verlag, 1985.

[49] P. Stevenhagen. The number field sieve. In *Algorithmic Number Theory: Lattices, Number Fields, Curves and Cryptography*, volume 44 of *MSRI Publications*, pages 83–100. Cambridge University Press, 2008.

[50] I. Stewart and D. O. Tall. *Algebraic Number Theory*. Chapman and Hall Mathematics Series. Chapman and Hall, 1987.

[51] W. Venables and B. Ripley. *Modern Applied Statistics with S*. Springer, fourth edition, 2002. http://cran.r-project.org/web/packages/MASS/index.html.

[52] Eric R. Verheul. Evidence that XTR is more secure than supersingular elliptic curve cryptosystems. *Journal of Cryptology*, 17(4):277–296, 2004.

[53] D. Wiedemann. Solving sparse linear equations over finite fields. *IEEE Transactions on Information Theory*, 32(1):54–62, 1986.

[54] P. Zimmermann and the GMP-ECM development team. *GMP-ECM: GMP Elliptic Curve Method*, 6.4.4 edition, 2013. http://ecm.gforge.inria.fr/.

Appendix A

Experimental Results

A.1 Polynomials f_2 Corresponding to Different $r(x)$ Values.

Joux-Pierrot polynomials f_1 and f_2 are generated by picking a polynomial $r(x)$ of degree < 12, all of whose coefficients are -1, 0 or 1 [30].

For each $r(x)$ of degree ≤ 3, we provide the number of non-zero terms in f_2, and the maximum absolute value of all the coefficients in f_2. Section 4.2.2 discuses how this information can be used to pick a good value of $r(x)$ to use for a specific prime p.

Note that for most choices of $r(x)$, there is a second choice which generates very similar values for f_1 and f_2. For more information, see Section 3.7.1.

$r(x)$	Number of terms in f_2	Largest absolute value of coefficient in f_2
0	5	36
-1	5	132
1	5	348
$-x$ x	15	216
$-x-1$ $x-1$	15	324
$-x+1$ $x+1$	15	696

$-x^2$ x^2	15	216
$-x^2-1$ x^2-1	15	324
$-x^2+1$ x^2+1	15	696
$-x^2-x$ $-x^2+x$	25	432
$-x^2-x-1$ $-x^2+x-1$	25	792
$-x^2-x+1$ $-x^2+x+1$	25	936
x^2-x x^2+x	25	540
x^2-x-1 x^2+x-1	25	504
x^2-x+1 x^2+x+1	25	1236
$-x^3$ x^3	14	216
$-x^3-1$ x^3-1	14	324
$-x^3+1$ x^3+1	14	696
$-x^3-x$ x^3+x	30	432
$-x^3-x-1$ x^3+x-1	30	648
$-x^3-x+1$ x^3+x+1	30	1080
$-x^3+x$ x^3-x	30	432
$-x^3+x-1$ x^3-x-1	30	648
$-x^3+x+1$ x^3-x+1	30	1080

$-x^3 - x^2$ $x^3 - x^2$	30	432
$-x^3 - x^2 - 1$ $x^3 - x^2 - 1$	30	648
$-x^3 - x^2 + 1$ $x^3 - x^2 + 1$	30	1080
$-x^3 - x^2 - x$ $x^3 - x^2 + x$	33	900
$-x^3 - x^2 - x - 1$ $x^3 - x^2 + x - 1$	34	1512
$-x^3 - x^2 - x + 1$ $x^3 - x^2 + x + 1$	34	1188
$-x^3 - x^2 + x$ $x^3 - x^2 - x$	33	828
$-x^3 - x^2 + x - 1$ $x^3 - x^2 - x - 1$	33	648
$-x^3 - x^2 + x + 1$ $x^3 - x^2 - x + 1$	33	1632
$-x^3 + x^2$ $x^3 + x^2$	30	432
$-x^3 + x^2 - 1$ $x^3 + x^2 - 1$	30	648
$-x^3 + x^2 + 1$ $x^3 + x^2 + 1$	30	1080
$-x^3 + x^2 - x$ $x^3 + x^2 + x$	34	1116
$-x^3 + x^2 - x - 1$ $x^3 + x^2 + x - 1$	34	864
$-x^3 + x^2 + x$ $x^3 + x^2 - x$	33	612
$-x^3 + x^2 + x - 1$ $x^3 + x^2 - x - 1$	33	1044
$-x^3 + x^2 + x + 1$ $x^3 + x^2 - x + 1$	33	1236

Table A.2: Data on the polynomials f_2 corresponding to different $r(x)$ values.

A.2 Percentiles of Norm Distribution with Different u Values

The following two tables list norm percentile data for BN curves using various values of u and $r(x)$. The percentile represents the base-10 logarithm of the product of the norms below which the given percent of norm products fall. See Section 4.2.3 for more information.

Percentage of norms	10%	25%	50%	75%	90%	100%
u_{min}	426.2	431.7	436.8	441.2	444.6	455.0
u_{mid}	426.7	432.1	437.2	441.6	445.0	455.8
u_{max}	427.1	432.5	437.6	442.0	445.4	456.5

Table A.3: Percentiles for BN product of norms for $r(x) = x^3$.

Percentage of norms	10%	25%	50%	75%	90%	100%
u_{min}	433.8	438.8	443.6	447.7	450.8	460.3
u_{mid}	434.3	439.3	444.1	448.1	451.2	460.7
u_{max}	434.6	439.7	444.5	448.6	451.6	461.7

Table A.4: Percentiles for BN product of norms for $r(x) = x^3 + x^2 + x - 1$.

A.3 Maximum Observed Norms

The following table lists the maximum observed absolute values of the norms N_{K_1} and N_{K_2} for each of the three BN curve experiments discussed in Sections 4.2.1 to 4.2.3. The maximum observed absolute value of the product of the norms is also listed.

u	$r(x)$	N_{K_1}	N_{K_2}	$N_{K_1}N_{K_2}$
6917529027641089837	x^3	8.796×10^{250}	1.328×10^{204}	6.820×10^{454}
	$-x^3$	8.796×10^{250}	2.598×10^{206}	7.990×10^{456}
6790547177159395210	0	7.174×10^{250}	9.826×10^{200}	5.512×10^{451}
	x^2+x+1	7.174×10^{250}	2.323×10^{211}	9.953×10^{461}
	x^3	7.174×10^{250}	5.498×10^{204}	1.122×10^{455}
	x^3+x^2-x-1	7.174×10^{250}	5.489×10^{210}	3.577×10^{461}
6332666225848389958	0	3.329×10^{250}	7.941×10^{200}	1.098×10^{451}
	x^3	3.329×10^{250}	8.541×10^{204}	1.094×10^{455}
	x^3+x^2+x-1	3.329×10^{250}	9.290×10^{209}	2.178×10^{460}
6931758979282357578	0	8.997×10^{250}	4.859×10^{200}	6.711×10^{451}
	x^3	8.997×10^{250}	6.646×10^{204}	5.708×10^{455}
	x^3+x^2+x-1	8.997×10^{250}	2.175×10^{210}	4.823×10^{460}
7530851732716310718	0	2.239×10^{251}	1.472×10^{201}	1.019×10^{452}
	x^3	2.239×10^{251}	2.364×10^{205}	3.536×10^{456}
	x^3+x^2+x-1	2.239×10^{251}	6.298×10^{210}	4.729×10^{461}

Table A.5: Maximum observed norms and norm products.

A.4 Output from the Box-Cox Method and Linear Model Summary

The following tables and figures are output from the R functions "boxcox" and "summary.lm" [44]. For more information, see Section 5.1.

Figure A.1 and Table A.6 give the output of the Box-Cox method for the K_1 norm.

λ	10	11	12	13	14
log-Likelihood	2101.31752	2775.61843	4991.08849	2958.15833	1822.86615

Table A.6: Selected log-Likelihoods for BN K_1 norms.

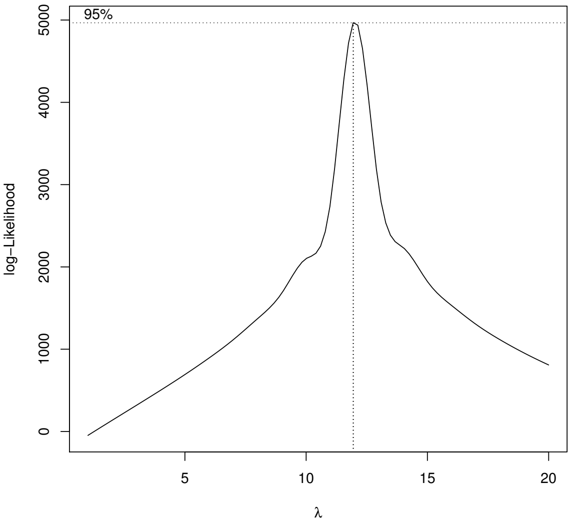

Figure A.1: Output of the Box-Cox method for BN K_1 norms.

Figure A.2 gives data relating to the fit of the model

$$Y^{12} = 0.001557 + 0.9980X$$

for the BN K_1 norm.

Figure A.3 and Table A.7 give the output of the Box-Cox method for the BN K_2 norm.

λ	46	47	48	49	50
log-Likelihood	2030.930	2032.743	2033.973	2034.646	2034.786
λ	51	52	53	54	55
log-Likelihood	2034.418	2033.564	2032.243	2030.476	2028.281

Table A.7: Selected log-Likelihoods for the BN K_2 norm.

Residuals:

Min	1Q	Median	3Q	Max
$-2.458e-03$	$6.455e-04$	$5.379e-05$	$7.083e-04$	$2.120e-03$

Coefficients:

	Estimate	Std. Error	t value	Pr($>|t|$)
(Intercept)	1.557e-03	6.054e-05	25.72	<2e-16
X	9.980e-01	1.048e-04	9524.49	<2e-16

Residual standard error: 0.0009565 on 998 degrees of freedom
Multiple R-squared: 1, Adjusted R-squared: 1
F-statistics: 9.072e+07 on 1 and 998 DF, p-value: <2.2e-16

Figure A.2: Summary results for linear model of BN K_1 norm.

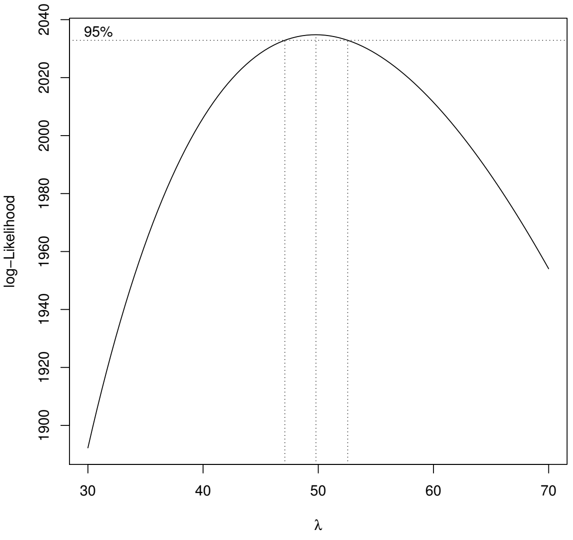

Figure A.3: Selected log-Likelihoods for the BN K_2 norm.