Size Optimization of Sextic Polynomials in the Number Field Sieve

Shi Bai, Paul Zimmermann

SIZE OPTIMIZATION OF SEXTIC POLYNOMIALS IN THE NUMBER FIELD SIEVE

SHI BAI AND PAUL ZIMMERMANN

Abstract. The general number field sieve (GNFS) is the most efficient algorithm known for factoring large integers. It consists of several stages, the first one being polynomial selection. The quality of the chosen polynomials in polynomial selection can be modelled in terms of size and root properties. In this paper, we describe some methods to optimize the size property of sextic polynomials.

1. Introduction to GNFS

The general number field sieve [10] is the most efficient algorithm known for factoring large integers. It has been used in many (current and previous) record factorizations such as RSA-768 [17]. GNFS consists of several stages including polynomial selection, sieving, filtering, linear algebra and finding square roots.

Let n be the integer to be factored. In polynomial selection, we want to choose two irreducible and coprime polynomials $f(x)$ and $g(x)$ over $\mathbb{Z}$ which share a common root m modulo n. In practice, the homogenized polynomials $F(x, y)$ and $G(x, y)$ are often used. We want to find many coprime pairs $(a, b) \in \mathbb{Z}^2$ such that the polynomials values $F(a, b)$ and $G(a, b)$ are simultaneously smooth. An integer is smooth with respect to bound B (or B-smooth) if none of its prime factors are larger than B. The line sieving and lattice sieving [16] are commonly used to identify such pairs (a, b). The running-time of sieving depends on the quality of the chosen polynomials in polynomial selection, hence many polynomial pairs will be generated and optimized in order to produce a good one.

This paper discusses algorithms for size optimization in polynomial selection in the number field sieve. We focus on polynomial selection with two polynomials, one of which is a linear polynomial and the other is a polynomial of degree six. Such polynomials are of great practical interest since they have been used in current and previous record factorizations such as RSA-768 [17] and may be used for future records.

2. Polynomial selection

For large integers, most methods for polynomial selection [3, 8, 9, 11, 12] in GNFS use a linear polynomial for $g(x)$ and a quintic or sextic polynomial for $f(x)$. The standard method to generate such polynomial pairs is to expand n in base-(m_1, m_2) so $n = \sum_{i=0}^{d} c_i m_1^i m_2^{d-i}$. The polynomial pair is given by $f(x) = \sum_{i=0}^{d} c_i x^i$ and $g(x) = m_2 x - m_1$.

2010 *Mathematics Subject Classification.* Primary 11Y05, 11Y16.

The running-time of sieving depends on the smoothness of the polynomial values $|F(a,b)|$ and $|G(a,b)|$. Let $\Psi(x, x^{1/u})$ be the number of $x^{1/u}$-smooth integers below x for some $u > 0$. The Dickman-de Bruijn function $\rho(u)$ [6] is often used to estimate the density of smooth numbers $\Psi(x, x^{1/u})$. It can be shown that

$$\lim_{x\to\infty} \frac{\Psi(x, x^{1/u})}{x} = \rho(u).$$

The Dickman-de Bruijn function satisfies the differential equation

$$u\rho'(u) + \rho(u-1) = 0, \quad \rho(u) = 1 \text{ for } 0 \le u \le 1.$$

It may be shown that ρ satisfies the asymptotic estimate

$$\log(\rho(u)) = -(1+o(1))u\log u \text{ as } u \to \infty.$$

For practical purposes, the frequency of smooth numbers can be approximated by the Canfield-Erdős-Pomerance theorem, which can be stated as follows (Corollary 1.3 from [7]).

Theorem 2.1. *For any fixed $\epsilon > 0$, we have*

$$\Psi(x, x^{1/u}) = xu^{-u(1+o(1))}$$

as $x^{1/u}$ and u tend to infinity, uniformly in the region $x \ge u^{u/(1-\epsilon)}$.

We want to choose the polynomials in a way such that it can produce many smooth polynomial values across the sieve region. This heuristically requires that the size of polynomial values is small in general. In addition, one can choose an algebraic polynomial $f(x)$ which has many roots modulo small prime powers. Then the polynomial values are likely to be divisible by small prime powers. This may increase the smoothness chance for polynomial values. We describe some methods [8, 12] to estimate and compare the quality of polynomials.

2.1. **Quality of polynomials.** The quality of the chosen polynomials in polynomial selection can be modelled in terms of size and root properties [12].

2.1.1. *Size property.* Let (a,b) be pairs of relatively prime integers in the sieving region Ω. For the moment, we assume that a rectangle sieving region is used where $|a| \le U$ and $0 < b \le U$. We also assume that polynomial values $|F(a,b)|$ and $|G(a,b)|$ behave like random integers of similar size. The number of sieving reports (coprime pairs that lead to smooth polynomial values) can be approximated by

$$\frac{6}{\pi^2} \iint_\Omega \rho\left(\frac{\log|F(x,y)|}{\log B}\right) \rho\left(\frac{\log|G(x,y)|}{\log B}\right) \mathrm{d}x\,\mathrm{d}y.$$

The multiplier $6/\pi^2$ accounts for the probability of a, b being relatively prime.

Since G is a linear polynomial, we may assume that $\log(|G(a,b)|)$ does not vary much across the sieving region. A simplified approximation to compare polynomials (ignoring the constant multiplier) is

$$\iint_\Omega \rho\left(\frac{\log|F(x,y)|}{\log B}\right) \mathrm{d}x\,\mathrm{d}y. \tag{2.1}$$

The base-(m_1, m_2) expansion [8, 9] can yield polynomials whose coefficients are $O(n^{1/(d+1)})$. The leading coefficients c_d and c_{d-1} are usually much smaller than

$n^{1/(d+1)}$. The coefficient c_{d-2} is slightly smaller than $n^{1/(d+1)}$. For such polynomials, it is often better to use a skewed sieving region where the sieving bounds for a, b have ratio s, while keeping the area of the sieving region $2U^2$. The sieving bounds become $|a| \le U\sqrt{s}$ and $0 < b \le U/\sqrt{s}$. Each monomial in the polynomial $F(a,b)$ is bounded by $|c_i|U^d s^{i-d/2}$.

In the integral (2.1), computing ρ is time-consuming, especially if there are many candidates. We can use some coarser approximations.

Since $\rho(u)$ is a decreasing function of u, we want to choose a polynomial pair such that the size of $|F(a,b)|$ (and $|G(a,b)|$) is small on average over all (a,b). This roughly requires that the coefficients of the polynomials are small in absolute value.

We can compare polynomials by the logarithmic average of polynomial values across the sieving region.

$$\log\left(\iint_{\Omega} |F(x,y)|\,\mathrm{d}x\,\mathrm{d}y\right).$$

For computational convenience, one can use the logarithmic L^2 norm for polynomial $F(x,y)$ by

$$\frac{1}{2}\log\left(\iint_{\Omega} F^2(x,y)\,\mathrm{d}x\,\mathrm{d}y\right). \tag{2.2}$$

The logarithmic L^2-norm is influenced by the skewness and the location of real roots. The integral in (2.2) can be expressed as a polynomial in the coefficients of $F(x,y)$.

One can also change the range and shape of the integral region (the domain Ω), while keeping the skewness. We consider a modified logarithmic L^2-norm defined by

$$\frac{1}{2}\log\left(s^{-d}\int_{-1}^{1}\int_{-1}^{1} F^2(xs,y)\,\mathrm{d}x\,\mathrm{d}y\right). \tag{2.3}$$

where s is the skewness of sieving region.

The logarithmic L^2-norm given in Equation (2.3) is defined on a square domain. One can also use a variant with elliptic domain. We change to polar coordinates where $x = r\cos\theta$ and $y = r\sin\theta$.

$$\frac{1}{2}\log\left(s^{-d}\int_{0}^{2\pi}\int_{0}^{1} F^2(s\cos\theta,\sin\theta)\,r^{2d+1}\,\mathrm{d}r\,\mathrm{d}\theta\right). \tag{2.4}$$

The logarithmic L^2-norm in Equation (2.3) is not exactly the same as the logarithmic L^2-norm in Equation (2.4), because the integrals are over different domains (ellipse and rectangle). They are both (but slightly different) approximations to the size of polynomials.

For sextic polynomial, the logarithmic L^2-norm in Equation (2.4) can be expressed as

$$\begin{aligned}\frac{1}{2}\log\Big(\frac{\pi}{7168}\big(&231\,\tilde{c}_0^2 + 42\,\tilde{c}_0\tilde{c}_2 + 14\,\tilde{c}_0\tilde{c}_4 + 10\,\tilde{c}_0\tilde{c}_6 + 21\,\tilde{c}_1^2 + 14\,\tilde{c}_1\tilde{c}_3\\ &+ 10\,\tilde{c}_1\tilde{c}_5 + 7\,\tilde{c}_2^2 + 10\,\tilde{c}_2\tilde{c}_4 + 14\,\tilde{c}_2\tilde{c}_6 + 5\,\tilde{c}_3^2 + 14\,\tilde{c}_3\tilde{c}_5\\ &+ 7\,\tilde{c}_4^2 + 42\,\tilde{c}_4\tilde{c}_6 + 21\,\tilde{c}_5^2 + 231\,\tilde{c}_6^2\big)\Big)\end{aligned} \tag{2.5}$$

where $\tilde{c_i} = c_i s^{i-d/2}$.

For a given norm defined in Equations (2.3) or (2.4), one should not only be able to estimate accurately that norm for a given skewness, but find the optimal skewness that gives the minimal norm.

2.1.2. *Root property.* If a polynomial $f(x)$ has many roots modulo small prime powers, the polynomial values may behave more smooth than random integers of about the same size. Boender, Brent, Montgomery and Murphy [2, 11, 12, 13] described some quantitative measures of this effect (root property).

Let p be a prime and $x \geq 0$ be an integer. We denote $cont_p(x)$ the exponent of the largest power of p dividing x and $cont_p(0) = \infty$. Let S be a set of uniformly distributed random integers. We denote $cont_p(S)$ the average p-valuation over elements of S.

For a fixed prime p, the expected p-valuation $cont_p(S)$ is

$$1 \cdot \left(\frac{1}{p} - \frac{1}{p^2}\right) + 2 \cdot \left(\frac{1}{p^2} - \frac{1}{p^3}\right) + \cdots = \frac{1}{p-1}.$$

In the number field sieve, we want to know the expected p-valuation of homogeneous polynomial values. Let $F(x, y)$ be an algebraic polynomial and $f(x)$ be its dehomogenized polynomial. We discuss the roots of $F(x, y)$. Let $p^k \mid F(a, b)$ for some coprime integers a, b and some integer k. Then there are two cases: either $p \nmid b$ and $f(a/b) \equiv 0 \pmod{p^k}$ or $p \mid b$ and $h(b/a) \equiv 0 \pmod{p^k}$ where $h(x) = x^d f(1/x)$.

In the first case, pairs (a, b) can be identified by $(a/b \pmod{p^k}, 1)$. They are referred to as the *affine* roots. $F(x, y) \pmod{p^k}$ can have p^k possible affine roots, each of which relates to $p^k - p^{k-1}$ equivalent (a, b) pairs.

For the second case, pairs (a, b) can be identified by $\big(1, b/a \pmod{p^k}\big)$. We call them the *projective* roots. There are at most p^{k-1} projective roots. Each relates to $p^k - p^{k-1}$ equivalent (a, b) pairs.

Let $n_{p,k}$ be the number of affine and projective roots (counting without multiplicities) of $F \pmod{p^k}$ for $k \geq 1$. The expected p-valuation of homogeneous polynomial values is

$$cont_p(F) = \frac{1}{p+1} \sum_{k=1}^{\infty} \frac{n_{p,k}}{p^{k-1}}. \tag{2.6}$$

Murphy [12] defines the $\alpha(F)$ function to compare the cumulative expected p-valuation of polynomial values to random integers of similar size. $\alpha(F)$ can be considered as the logarithmic benefit compared to using random integers.

$$\alpha(F) = \sum_{p \leq B} \left(\frac{1}{p-1} - cont_p(F)\right) \log p.$$

In the number field sieve, we want $\alpha(F)$ negative and large in absolute value.

2.1.3. *Combined score function.* The logarithmic L^2-norm in Equation (2.3) can be modified to take the root property into account. Since the $\alpha(F)$ function affects the polynomial size on logarithmic scale, the combined function can be defined by adding $\alpha(F)$ to the logarithmic L^2-norm. We refer to it as the combined score function. The combined score is only a rough estimate to compare polynomials. In practice, it is only trustful when the differences between polynomials are large.

2.2. **Optimizing the quality of polynomials.** Polynomial selection can be divided into three steps: polynomial generation, size optimization and root optimization.

In polynomial generation, we generate many raw polynomials whose size is admissible. We further reduce the size of the raw polynomials in size optimization. Many polynomials can have comparable size after size optimization. We produce and choose the best polynomials in terms of root properties in root optimization.

Translation and rotation are useful to optimize the size and root properties. Let $f(x) = \sum_{i=0}^{d} c_i x^i$ and $g(x) = m_2 x - m_1$ where $m_1/m_2 \pmod n$ is the common root.

Translation of $f(x)$ by k gives a new polynomial $f_k(x)$ defined by $f_k(x) = f(x + k)$. The linear polynomial $g_k(x)$ is $m_2 x - m_1 + k m_2$. The common root becomes $m_1/m_2 - k \pmod n$. Translation does not alter the root properties.

Rotation by a polynomial $\lambda(x)$ gives a new polynomial $f_{\lambda(x)}(x)$ defined by $f_{\lambda(x)}(x) = f(x) + \lambda(x)\, g(x)$. The linear polynomial is unchanged $g_{\lambda(x)}(x) = g(x) = m_2 x - m_1$. The root is unchanged. $\lambda(x)$ is often a linear or quadratic polynomial, depending on n and on the skewness of $f(x)$. Rotation can affect both size and root properties.

3. Size optimization

Polynomial generation (e.g. using Kleinjung's methods [8, 9]) gives many raw polynomials with small leading coefficients. The raw polynomials have very small $|c_d|, |c_{d-1}|$ and small $|c_{d-2}|$. The coefficients $|c_{d-3}|, \cdots, |c_0|$ are comparable to $(n/c_d)^{1/d}$.

For quintic polynomials, coefficients $|c_5|, |c_4|$ and $|c_3|$ are small. The next non-controlled coefficient is c_2. Let the sieving bounds be $|a| \le U\sqrt{s}$ and $0 < b \le U/\sqrt{s}$. The polynomial values are bounded below by $|c_2| s^{-1/2} U^5$. As $s \ge 1$, the contribution of c_2 on the polynomial value is already reduced by a factor of $s^{-1/2}$.

For sextic polynomials, the polynomial values are bounded below by terms $|c_3| U^6$. As c_3 is not controlled in the polynomial generation step, we do not get a reduction in size like the $s^{-1/2}$ factor for quintic polynomials. Therefore, it is important to size-optimize them before trying to optimize the root properties.

In this paper, we focus on the size optimization of raw, sextic polynomials. In size optimization, we want to produce polynomials with smaller logarithmic L^2-norm (e.g. Equation (2.4)) by changing the skewness, translating and rotating.

3.1. **Local descent method.** Let $f(x)$ be a sextic polynomial. We can use quadratic rotations since $c_3, \cdots, c_0$ have order $(n/c_d)^{1/d}$. A quadratic rotation is defined by

$$f_{u,v,w}(x) = f(x) + (ux^2 + vx + w)\, g(x) \tag{3.1}$$

for some integers u, v, w.

Murphy [12] used the classic multivariable optimization technique to optimize the L^2-norm. For sextic polynomials, there are five variables u, v, w, k, s, where k is the translation amount and s is the skewness. u, v, w, k are integers and s is real.

The allowed range of these parameters is huge. The standard iterative methods, such as the gradient descent, are slow and tend to get stuck in local minima. For efficiency, we use a local descent method to optimize the size.

In each iteration, we attempt some translations k and rotations u, v, w, and descend into the local minimum in the direction determined by some k, u, v, w. During the procedure, we need to re-optimize the skewness of the polynomial. We describe the method in Algorithm 1.

Algorithm 1: Local descent method

Input : polynomial pair $f(x) = \sum_{i=0}^{d} c_i x^i$ and $g(x) = m_2 x - m_1$;
Output: polynomial pair f', g' of smaller L^2-norm;

1 $k = u = v = w = 1$;
2 **while** *local minimum is found or loop limit is reached* **do**
3 $f'(x) = f(x \pm k), g'(x) = g(x) \pm k m_2$;
4 **if** *either* $L^2(f') < L^2(f)$ **then**
5 $f = f',\ g = g',\ k = 2k$;
6 **else**
7 $k = \lceil k/2 \rceil$;
8 $f'(x) = f(x) \pm u\, x^2\, g(x)$;
9 **if** *either* $L^2(f') < L^2(f)$ **then**
10 $f = f',\ u = 2u$;
11 **else**
12 $u = \lceil u/2 \rceil$;
13 Search similarly (e.g. lines 8-12) for linear and constant rotations;
14 **return** $f(x), g(x)$;

The method seems to work for quintic polynomials, when the searching space is not too huge. However, it performs badly in practice for sextic polynomials. Many iterations get stuck at local minima without giving much reduction in size. We demonstrate this situation below.

We examine a data set consisting of 10^5 raw sextic polynomials for RSA-768. The polynomials are generated by Kleinjung's 2008 algorithm [9]. Figure 1 shows the discrete density distribution of logarithmic L_2-norm for the raw and optimized (by the local descent) polynomials.

In Figure 1, the raw polynomials have average logarithmic L^2-norm 80.75 and standard deviation 1.00. The optimized polynomials have average logarithmic L^2-norm 79.06 and standard deviation 3.55. It can be seen that only a few polynomials are optimized well by the local descent procedure. Many of them seem to descend to a local minimum rapidly and then get stuck. We discuss below some better methods to optimize such polynomials.

To overcome local minima, we could use some global optimization methods such as simulated annealing. However, they do not seem to work efficiently in our experiments, due to the huge search space and large coefficients.

Instead, we first translate the algebraic polynomial to increase the skewness. Heuristically, it moves away from the starting point and decreases the chance to get stuck in a local minimum. If the skewness of the polynomial is larger than the translation amount k, the translation does not affect the norm significantly. This can be seen from the coefficients of $f(x + k)$. A local optimization method such as descent can then be applied. One question is how to decide the translation amount. We describe some methods in Subsection 3.2.

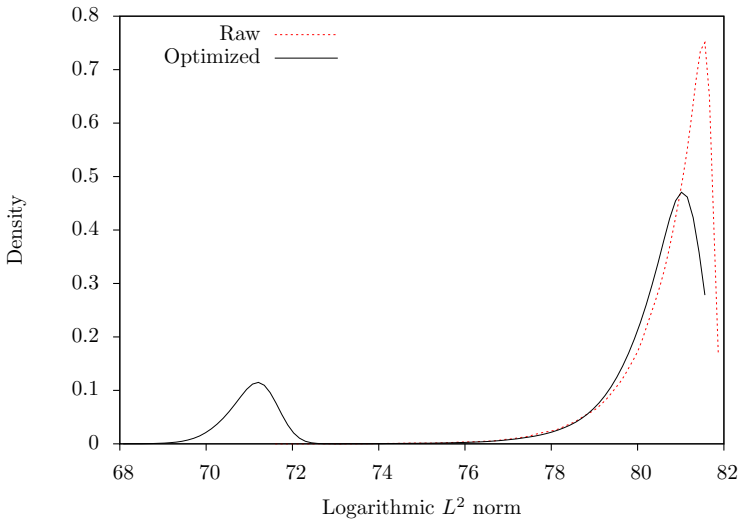

FIGURE 1. Local descent optimization

3.2. **A better method.** We want to produce a polynomial with small L^2-norm by translation and rotation.

In the raw polynomial, c_0, c_1, c_2, c_3 have similar size and are much larger than c_4, c_5, c_6. In Equation (2.5), the $\tilde{c}_0, \tilde{c}_1, \tilde{c}_2$ are bounded by $\tilde{c}_3$. Therefore, the L^2-norm can be controlled by terms involving $\tilde{c}_3, \tilde{c}_4, \tilde{c}_6$. A lower bound, not depending on skewness, is dominated by the term $\tilde{c}_3^2 = c_3^2$. We demonstrate this situation for a raw polynomial A_{768} in Appendix A. It is a raw polynomial generated by Kleinjung's 2008 algorithm [9] that could be used for RSA-768.

Let $s = 3916800$ be the optimal skewness for the raw polynomial. We consider the relative weight of each term in Equation (2.5). The largest term is $5\tilde{c}_3^2 \approx 2.58 \times 10^{66}$, whereas the second largest term is $10\,\tilde{c}_2\tilde{c}_4 \approx 1.23 \times 10^{61}$. Hence, a small c_3 is a necessary condition for a small L^2-norm. The idea is to minimize c_3 by translation.

Translation by k gives a polynomial in x whose coefficients are functions of k:

$$\begin{aligned} f(x+k) = {} & c_6x^6 \\ & + (6c_6k + c_5)x^5 \\ & + (15c_6k^2 + 5c_5k + c_4)x^4 \\ & + (20c_6k^3 + 10c_5k^2 + 4c_4k + c_3)x^3 \\ & + \cdots \end{aligned}$$

Let $c_i(k)$ be the coefficients of the i-th term in the translated polynomial. $c_3(k)$ of $f(x+k)$ is a cubic polynomial in k. The coefficients $c_0(k), c_1(k), c_2(k)$ will increase due to translation. We can use rotation to reduce them, if needed.

3.2.1. *Minimizing $c_3(k)$.* The cubic polynomial $c_3(k)$ has either one or three real roots. For each real root r, we choose K to be either $\lceil r \rceil$ or $\lfloor r \rfloor$, whichever minimizes

$|c_3(k)|$. We translate $f(x)$ by K. The optimization is expected to work for all sextic polynomials since there exists at least one real root for a cubic polynomial.

In the cubic polynomial $c_3(k)$, the constant term c_3 is $O(m_1)$ (see Lemma 2.1 of [8]). The real root r is about $O((m_1/c_6)^{1/3})$. Hence $c_5(K)$ is bounded by $O(m_1^{1/3}c_6^{2/3} + c_5)$ and $c_4(K)$ is bounded by $O(m_1^{2/3}c_6^{1/3} + c_4)$. Empirically, c_4 is comparable to $c_4(K)$ for the raw polynomials found by algorithms [8, 9]. On the other hand, $m_1 \gg |c_6|$ and $|c_6| \approx |c_5|$ and hence the coefficient $c_5(K)$ can increase significantly.

After translation, $c_3(k)$ is minimized and often smaller than the original c_3. Let $\delta = K - r$ and hence $|\delta| < 1$. It follows that

$$\begin{aligned} |c_3(K)| &= |20c_6K^3 + 10c_5K^2 + 4c_4K + c_3| \\ &= |20c_6(\delta^3 + 3r^2\delta + 3r\delta^2) + 10c_5(2r\delta + \delta^2) + 4c_4\delta| \\ &\le 20\,|c_6|\,(1 + 3r^2 + 3\,|r|) + 10\,|c_5|\,(2\,|r| + 1) + 4\,|c_4| \end{aligned} \tag{3.2}$$

Given $r \sim O((m_1/c_6)^{1/3})$ where $m_1 \gg |c_6|$ and $|c_5| \approx |c_6|$, equation (3.2) above has order $O(m_1^{2/3}c_6^{1/3} + c_4)$. $|c_3(K)|$ is likely to be smaller than $|c_3| \sim O(m_1)$ since $c_4 < c_3$ in the raw polynomial. Assume further that $|c_4| \sim O(m_1^{2/3}c_6^{1/3})$, which appears to be practical (see Kleinjung's 2008 method [9]). After translation, $|c_3|$ can be reduced by a factor of $(m_1/c_6)^{1/3}$.

Once K is fixed in minimizing $c_3(k)$, we can further optimize the polynomial locally by the local descent method.

In the translated polynomial $f_K(x)$, the coefficients $c_5(K) \sim O(m_1^{1/3}c_6^{2/3})$, $c_4(K) \sim O(m_1^{2/3}c_6^{1/3})$, $c_3(K) \sim O(m_1^{2/3}c_6^{1/3})$, $c_2(K) \sim O(m_1^{4/3}/\,c_6^{1/3})$, $c_1(K) \sim O(m_1^{5/3}/\,c_6^{2/3})$, $c_0(K) \sim O(m_1^2/c_6)$. The coefficients $c_2(K), c_1(K), c_0(K)$ are increased during the translation. We can reduce them using rotation in the local optimization. As an example, we apply a quadratic rotation on $f_K(x)$ to reduce $c_0(K), c_1(K), c_2(K)$ to $O(m_1)$. The quadratic polynomial $ux^2 + vx + w$ used in the rotation has parameters $w \sim O(m_1/c_6), v \sim O((m_1/c_6)^{2/3})$ and $u \sim O((m_1/c_6)^{1/3})$ (using $m_2 \ll m_1$). Let the rotated polynomial be $\tilde{f}_K(x)$ whose coefficients are $\tilde{c}_i(K)$ for $0 \le i \le 6$. The coefficient $\tilde{c}_3(K) \sim O(m_1^{2/3}c_6^{1/3} + m_2(m_1/c_6)^{1/3}) \sim O(m_1^{2/3}c_6^{1/3})$. Hence $c_0(K), c_1(K), c_2(K)$ are reduced to $O(m_1)$ without increasing too much $c_3(K)$. Compared $\tilde{f}_K(x)$ to the raw polynomial, $\tilde{c}_5(K)$ is increased, while $\tilde{c}_3(K)$ is often smaller. If the gain from a smaller $\tilde{c}_3(K)$ exceeds the deterioration from a larger $\tilde{c}_5(K)$, the L^2-norm can be reduced. In practice, the local descent method (Algorithm 1) can be applied, instead of a single rotation.

3.2.2. *Example and statistics.* We give an example for the polynomial A_{768} in Appendix A. It has logarithmic L_2-norm 72.59. The coefficient of x^3 in $f(x + k)$ is

$$\begin{aligned} &71727600k^3 + 190647000k^2 + 112950493882223418033937 2k + \\ &7186937011302402252746128141881 42. \end{aligned}$$

The cubic polynomial has a real root near $k = -191352410$. We translate $f(x)$ by k and then apply the local descent method to the translated polynomial. The resulting optimized polynomial B_{768} in Appendix A has logarithmic L_2-norm 67.60.

The method works better on average than the local descent method used alone. We consider the same data set of 10^5 polynomials used in Figure 1. Figure 2 shows

the discrete density distribution of logarithmic L_2-norm for the raw and optimized polynomials. The improved method can reduce the average logarithmic L_2-norm to 70.34 with a standard deviation 0.60. We gain almost 9 on the average logarithmic L_2-norm with respect to the local descent method used alone.

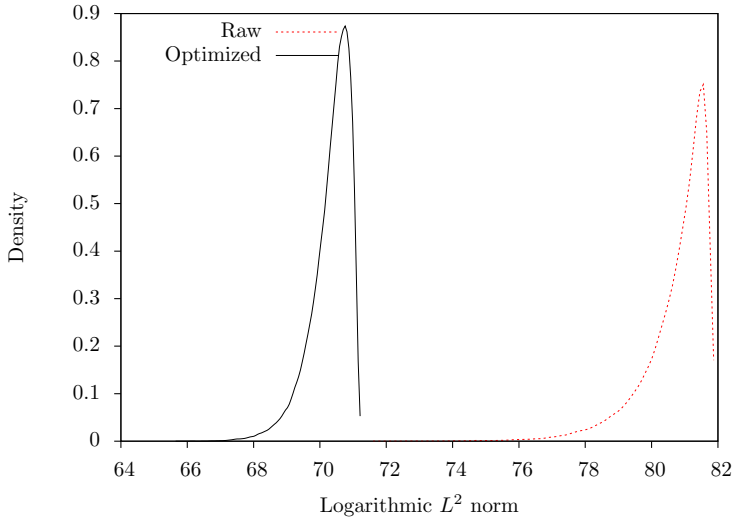

FIGURE 2. Optimizing $c_3(k)$ before the local descent

3.2.3. *Further improvement.* Thorsten Kleinjung (personal communication) describes a method which helps further reduce the norm. Before translation, we attempt several cubic rotations by $f(x) + \delta x^3 g(x)$ for small δ's on the raw polynomial $f(x)$. This gives some variation during the optimization. For each rotated polynomial, we repeat the optimization procedure and record the minimum norm found.

The variation gives some benefits in practice. We consider the same data set of 10^5 polynomials used in Figure 1. In experiments, we rotate polynomials by $|\delta| \leq 256$ and optimize the size using the above method. Figure 3 shows the discrete density distribution of logarithmic L_2-norm for the raw and optimized polynomials. This method can further reduce the average logarithmic L_2-norm to 69.84 with a standard deviation 0.56. We gain another 0.5 on the average logarithmic L_2-norm compared to the above method.

3.3. **Trade-off between size and root.** The raw polynomial often has a small c_5, which permits a larger rotation bound in root optimization. The size optimization procedure leads to a much larger c_5 due to translation. This may lead to a smaller rotation space. We could have optimized the root property (of the raw polynomial) first and then optimized the size by translating and changing the skewness. If the root property is outstanding, we might expect that it can better than the size-root (in order) optimization. However, we give a heuristic argument that this is difficult in practice.

Let $f_{u,v,w}(x)$ be the rotated polynomial in Equation (3.1). If $c_3 \sim O(m_1)$ and $c_0 \sim c_3 s^3$, $c_1 \sim c_3 s^2$, $c_2 \sim c_3 s$, we have an upper bound $O(s^6)$ for the rotation

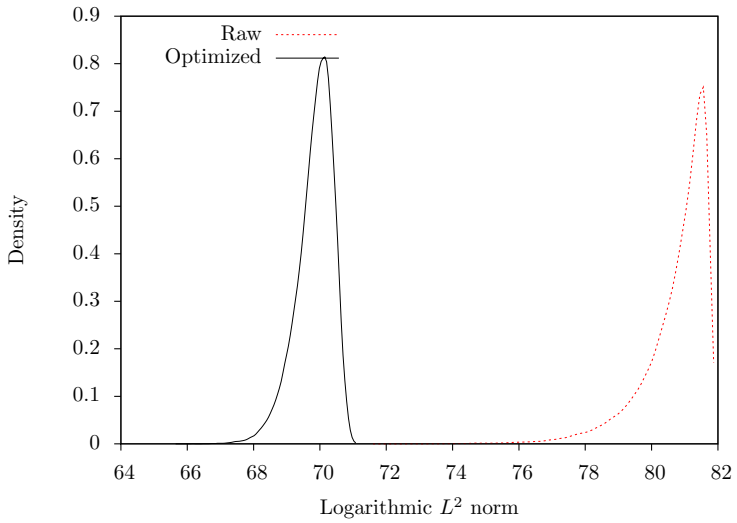

FIGURE 3. Optimizing $c_3(k)$ before the local descent: a better variant

space. We want to estimate the expected minimum $\alpha(F)$ after K polynomials are chosen where K is about s^6.

3.3.1. *Expected minimum* $\alpha(F)$. Emmanuel Thomé described a method (personal communication) to estimate the expected minimum of $\alpha(F)$ using order statistics. We assume that the $\alpha(F)$ values of random polynomials follow a standard Gaussian (normal) distribution $N(\mu, \sigma^2)$ (see Figure 4).

Let $\Phi(x)$ be the cumulative density function for the standard $N(0,1)$ normal distribution $\phi(x)$ where

$$\phi(x) = \frac{1}{\sqrt{2\pi}}\, e^{-x^2/2}, \quad \Phi(x) = \frac{1}{2}\left(1 + \operatorname{erf}\left(\frac{x}{\sqrt{2}}\right)\right).$$

In $\Phi(x)$, $\operatorname{erf}(x)$ is the error function [5] defined by

$$\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2}\, \mathrm{d}t.$$

The probability distribution for the minimum order statistic is given by

$$p_K(x) = K\ (1 - \Phi(x))^{K-1}\ \phi(x)$$

where K is the cardinality of the sample set. We use an asymptotic approximation [4] for the expected value of the minimum order statistic of the normal distribution.

$$\mu - \sigma\left(\sqrt{2\log K} - \frac{\log(\log K) + 1.377}{2\sqrt{2\log K}}\right). \tag{3.3}$$

In practice, we need to estimate the parameters μ, σ of the actual distribution. We examine a data set of 10^7 polynomials for RSA-768. The polynomials are generated by CADO-NFS [1] and Msieve [14, 15]. The data has mean $\mu = -0.257$ and standard deviation $\sigma = 0.824$. Here the average $\alpha(F)$ is negative since the

raw polynomials are generated in a way such that they are expected to have good projective root property (e.g. c_d is divisible by many small primes).

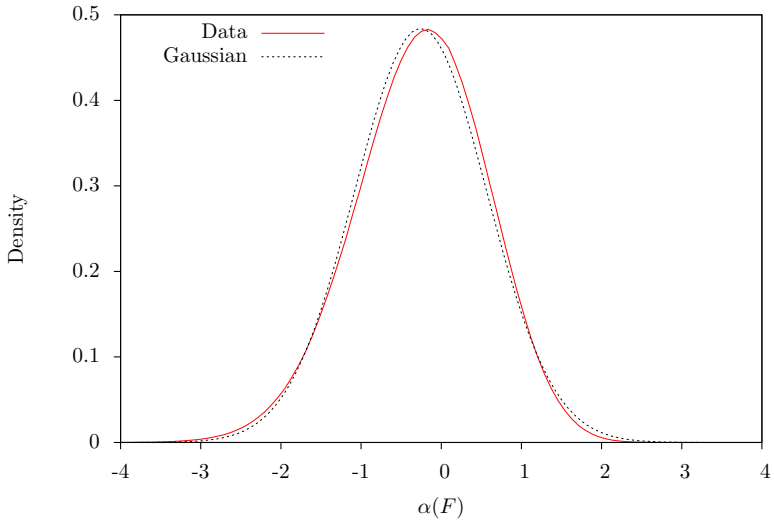

FIGURE 4. Distribution of α of raw RSA-768 polynomials

In Figure 4, we show the density estimate of the data. The estimated distribution of $\alpha(F)$ is close to a Gaussian distribution with above parameters μ, σ. We use these parameters to estimate the expected value of the minimum order statistic (e.g. Equation (3.3)).

3.3.2. *Order of optimization.* Assume $\alpha(F)$ follows a normal distribution with mean $\mu = -0.257$ and deviation $\sigma = 0.824$. Equation (3.3) shows that the expected minimum α is roughly proportional to the square root of logarithmic scale of skewness.

We consider the situation of sextic polynomials for RSA-768. Let $s = 10^{10}$, which is reasonably large for raw polynomials. It gives an expected minimum $\alpha = -13.80$ after s^6 polynomials are generated. In an ideal situation, we can expect to find such α without affecting the size. In practice, a rotation space of s^6 is very likely to increase the size and it is very hard to find polynomials with such α while keeping the size constrained. We can also apply a size optimization of two variables (translation and skewness) afterwards. However, such optimization is restricted as none rotation can be used and is likely to be ineffective.

On the other hand, if we first conduct size optimization, Figure 3 shows that a reduction of 10 in norm is common. A following root optimization can further reduce the combined norm by 7–11, despite increasing the size. Put together, size-root (in order) optimization often behaves much better than root-size optimization in experiments. Therefore, it is suggested to optimize the size property first and then the root property.

4. Conclusion

We discussed the size optimization in polynomial selection for the number field sieve. Size optimization aims to further reduce the size of the raw polynomials

by changing skewness, translating and rotating. Traditional local optimization techniques fail for many sextic polynomials when the integers to be factored are large. We described some better methods to optimize the size by determining an appropriate initial polynomial for the iteration and then locally optimizing the polynomial.

Appendix A. Some polynomials

The appendix contains polynomial pairs A_{768}, B_{768} discussed in the paper.

Polynomial A_{768}:

$$\begin{aligned} f(x) = \quad & 3586380\,x^6 \\ & + 19064700x^5 \\ & + 28237623470555854508484 3x^4 \\ & + 71869370113024022527461281418814 2x^3 \\ & + 43402001628933397612599912223809112 82x^2 \\ & - 12541568233611627968693736065307030120x \\ & + 90083741744675634459369471396413328 77 \\ g(x) = \quad & 53362054832582019225383\,x \\ & - 26457722251514149087911384249044520830 \end{aligned}$$

skewness: 3916800.00

L^2-norm: 72.59

$\alpha(F)$: -1.08

Polynomial B_{768}:

$$\begin{aligned} f(x) = \quad & 3586380\,x^6 \\ & - 4117247962908300x^5 \\ & + 22518332252355341901098 43x^4 \\ & + 13622093004046967078413861051 6x^3 \\ & - 17501466895317212327777571690641007037x^2 \\ & - 26110703038255899947787642868830402747 6731x \\ & + 76155151602800397749280550197763110360483636576 12 \\ g(x) = \quad & 5336205483258201922538 3\,x \\ & - 26457732461661641051994527730477303005 \end{aligned}$$

skewness: 2593792.00

L^2-norm: 67.60

$\alpha(F)$: -2.04

References

1. S. Bai, P. Gaudry, A. Kruppa, F. Morain, L. Muller, E. Thomé, P. Zimmermann, and et al. CADO-NFS v1.1, an implementation of the number field sieve. `http://cado-nfs.gforge.inria.fr`, 2011.
2. H. Boender. *Factoring large integers with the quadratic sieve*. PhD thesis, Leiden University, 1997.
3. J. Buhler, H. Lenstra, and C. Pomerance. Factoring integers with the number field sieve. In Lenstra and Lenstra [10], pages 50–94.
4. H. Cramér. *Mathematical Methods of Statistics*. Princeton University Press, 1999.
5. W. Gautschi. Chapter 7. Error function and Fresnel integrals. In M. Abramowitz and I. A. Stegun, editors, *Handbook of Mathematical Functions: with Formulas, Graphs, and Mathematical Tables*, page 1046. Dover Publications, 1972.
6. A. Granville. Smooth numbers: computational number theory and beyond. In *Proc. MSRI Conf. Algorithmic Number Theory: Lattices, Number Fields, Curves and Cryptography*. MSRI Publications, Volume 44, 2008.
7. A. Hildebrand and G. Tenenbaum. Integers without large prime factors. *Journal de Théorie des Nombres de Bordeaux*, 5(2):411–484, 1993.
8. T. Kleinjung. On polynomial selection for the general number field sieve. *Mathematics of Computation*, 75(256):2037–2047, 2006.
9. T. Kleinjung. Polynomial selection. In *CADO workshop on integer factorization*, INRIA Nancy, 2008. `http://cado.gforge.inria.fr/workshop/slides/kleinjung.pdf`.
10. A. K. Lenstra and H. W. Lenstra, Jr., editors. *The Development of the Number Field Sieve*, volume 1554 of *Lecture Notes in Mathematics*. Springer, 1993.
11. B. A. Murphy. Modelling the Yield of Number Field Sieve Polynomials. In *Algorithmic Number Theory - ANTS III, LNCS 1443*, pages 137–147, 1998.
12. B. A. Murphy. *Polynomial selection for the number field sieve integer factorisation algorithm*. PhD thesis, The Australian National University, 1999.
13. B. A. Murphy and R. P. Brent. On quadratic polynomials for the number field sieve. In *Proceedings of the CATS '98*, volume 20 of *Australian Computer Science Communications*, pages 199–213. Springer, 1998.
14. J. Papadopoulos. Call for volunteers: RSA768 polynomial selection, 2011. `http://www.mersenneforum.org/showthread.php?t=15540`.
15. J. Papadopoulos. Msieve v1.48, 2011. `http://sourceforge.net/projects/msieve/`.
16. J. M. Pollard. The lattice sieve. In Lenstra and Lenstra [10], pages 43–49.
17. T. Kleinjung, K. Aoki, J. Franke, A. K. Lenstra, E. Thomé, J. W. Bos, P. Gaudry, A. Kruppa, P. L. Montgomery, D. A. Osvik, H. J. J. te Riele, A. Timofeev, and P. Zimmermann. Factorization of a 768-bit RSA modulus. In *Proceedings of CRYPTO '10*, volume 6223 of *Lecture Notes in Computer Science*, pages 333–350. Springer, 2010.

Australian National University, Canberra, Australia.
E-mail address: `shi.bai@anu.edu.au`

INRIA Nancy, Grand Est, Villers-les-Nancy, France.
E-mail address: `Paul.Zimmermann@loria.fr`