On the amount of sieving in factorization methods

Ekkelkamp, W.H.

On the Amount of Sieving in Factorization Methods

Proefschrift

ter verkrijging van
de graad van Doctor aan de Universiteit Leiden,
op gezag van Rector Magnificus prof.mr. P.F. van der Heijden,
volgens besluit van het College voor Promoties
te verdedigen op woensdag 20 januari 2010
klokke 13.45 uur
door

Willemina Hendrika Ekkelkamp

geboren te Almelo
in 1978

Samenstelling van de promotiecommissie:

promotoren:	Prof.dr. R. Tijdeman
	Prof.dr. A.K. Lenstra (EPFL, Lausanne, Zwitserland)
copromotor:	Dr.ir. H.J.J. te Riele (CWI)
overige leden:	Prof.dr. R.J.F. Cramer (CWI / Universiteit Leiden)
	Prof.dr. H.W. Lenstra
	Prof.dr. P. Stevenhagen
	Dr. P. Zimmermann (INRIA, Nancy, Frankrijk)

The research in this thesis has been carried out at the Centrum Wiskunde & Informatica (CWI) and at the Universiteit Leiden. It has been financially supported by the Netherlands Organisation for Scientific Research (NWO), project 613.000.314.

On the Amount of Sieving in Factorization Methods

Ekkelkamp, W.H., 1978 –
On the Amount of Sieving in Factorization Methods
ISBN:

Thomas Stieltjes Institute
for Mathematics

The illustration on the cover is a free interpretation of line and lattice sieving in combination with the amount of relations found.

Contents

Chapter 1

Introduction

1.1 Factoring large numbers

In 1977 Rivest, Shamir, and Adleman introduced the RSA public-key cryptosystem [44]. The safety of this cryptosystem is closely related to the difficulty of factoring large integers. It has not been proved that breaking RSA is equivalent with factoring, but it is getting close: see, for instance, the work of Aggarwal and Maurer [1]. Nowadays RSA is widely used, so factoring large integers is not merely an academic exercise, but has important practical implications.

The searches for large primes and for the prime factors of composite numbers have a long history. Euclid ($\pm$ 300 B.C.) was one of the first persons to write about compositeness of numbers, followed by Eratosthenes (276–194 B.C.), who came up with an algorithm that finds all primes up to a certain bound, the so-called Sieve of Eratosthenes. For an excellent overview of the history of factorization methods, we refer to the thesis of Elkenbracht-Huizing ([18], Ch. 2).

Over the years, people came up with faster factorization methods, when the factors to be found are large. Based on ideas of Kraitchik, and Morrison and Brillhart, Schroeppel introduced the linear sieve (1976/1977). With a small modification by Pomerance this became the quadratic sieve; albeit a very basic version. Improvements are due to Davis and Holdridge, and Montgomery (cf. [15], [39], [45]). Factoring algorithms based on sieving are much faster than earlier algorithms, as the expensive divisions are replaced by additions. The expected running time of the quadratic sieve, based on heuristic assumptions, is

$$L(N) = \exp((1 + o(1))(\log N)^{1/2}(\log\log N)^{1/2}), \text{ as } N \to \infty,$$

where N is the number to be factored and the logarithms are natural [14]. An improvement of the quadratic sieve is the multiple polynomial quadratic sieve.

The same idea of sieving is used in the number field sieve, based on ideas of Pollard (pp. 4–10 of [27]) and further developed by H.W. Lenstra, among others.

The heuristic expected running time of the number field sieve is given by (cf. [27])

$$L(N) = \exp(((64/9)^{1/3} + o(1))(\log N)^{1/3}(\log\log N)^{2/3}), \text{ as } N \to \infty.$$

Asymptotically, the number field sieve is the fastest known general method for factoring large integers, and in practice for numbers of about 90 and more decimal digits. We will go into the details of the multiple polynomial quadratic sieve and the number field sieve in the next two sections, as we use them for experiments in this thesis. Sections 1.4, 1.5, and 1.6 contain an overview of the thesis.

1.2 Multiple polynomial quadratic sieve

The multiple polynomial quadratic sieve (MPQS) is based on the quadratic sieve, so we start with a short description of the basic quadratic sieve (QS). Take N as the (odd) composite number (not a perfect power) to be factored. Then QS searches for integers x and y that satisfy the equation

$$x^2 \equiv y^2 \mod N.$$

If this holds, and $x \not\equiv y \mod N$ and $x \not\equiv -y \mod N$, then $\gcd(x-y, N)$ is a proper factor of N.

The algorithm starts with the polynomial $f(t) = t^2 - N$, $t \in \mathbb{Z}$. For every prime factor p of $f(t)$ the congruence $t^2 \equiv N \mod p$ must hold. For odd primes p this can be expressed as the condition $(\frac{N}{p}) = 1$ (Legendre symbol).

A number is called F-smooth if all its prime factors are below some given bound F. These numbers are collected in the sieving step of the algorithm. The factorbase FB is defined as the set of -1 (for practical reasons), 2, and all the odd primes p up to a bound F for which $(\frac{N}{p}) = 1$. Now we evaluate $f(t)$ for all integer values $t \in [\sqrt{N} - M, \sqrt{N} + M]$, where M is subexponential in N, and keep only those for which $|f(t)|$ is F-smooth. We can write this as

$$t^2 \equiv \prod_{p \in FB} p^{e_p(t)} \mod N, \tag{1.1}$$

where $e_p(t) \geq 0$ and we call expression (1.1) a relation.

The left-hand side is a square. Hence the product of such expressions is also a square. If we generate relations until we have more distinct relations than primes in the factorbase, there is a non-empty subset of this set of relations such that multiplication of the relations in this subset gives a square on the right-hand side. To find a correct combination of relations, we use linear algebra to find dependencies in a matrix with the exponents $e_p(t) \mod 2$ (one relation per row). Possible algorithms for finding such a dependency include Gaussian elimination, block Lanczos, and block Wiedemann; the running time of Gaussian elimination is $O(n^3)$ and the running time of block Lanczos and block Wiedemann is $O(nW)$, where n is the dimension of the matrix and W the weight of the matrix ([12], [13], [33], [49]).

Once we have a dependency of k relations for $t = t_1, t_2, \ldots, t_k$, we set $x = t_1 t_2 \ldots t_k \mod N$ and

$$y = \prod_{p \in FB} p^{\frac{e_p(t_1)+\cdots+e_p(t_k)}{2}} \mod N.$$

Then $x^2 \equiv y^2 \pmod{N}$. Now we compute $\gcd(x-y, N)$ and hopefully we have found a proper factor. If not, we continue with the next dependency. As the probability of finding a factor is at least $1/2$ for each dependency (as N has at least two distinct prime factors), we are in practice almost certain to have found the prime factors after having computed a small number of gcd's as above.

To make clear where the term 'sieving' from the title comes from, we take a closer look at the polynomial $f(t)$. If we know that 2 is a divisor of $f(3)$, then 2 is a divisor of $f(3+2k)$, for all $k \in \mathbb{Z}$; more generally, once we have located a t-value for which a given prime p divides $f(t)$, we know a residue class modulo p of t-values for which p divides $f(t)$. This can be exploited as follows. Initialize an array a as $\log(|f(t)|)$ for all integers $t \in [\sqrt{N} - M, \sqrt{N} + M]$, so we have $a[i] = \log(|f(i)|)$ for $i \in [\sqrt{N} - M, \sqrt{N} + M]$ for some appropriate M. Then for all primes p in the factorbase locate the positions i where p divides $f(i)$ and subtract $\log(p)$ from the numbers on these positions in the array (if a prime power divides a certain position, subtract the corresponding multiple of $\log(p)$). After the sieving we only pick out those positions in the array that are zero, as these represent the F-smooth values. Note that this is an idealized situation: in practice rounding will take place and has to be taken care of.

The polynomial values increase quadratically with the length of the sieving interval. To handle this problem, Davis and Holdridge came with the following approach [15]. Choose a so-called special prime $q_0 > F$ with $(\frac{N}{q_0}) = 1$, and choose r_0 with $r_0^2 \equiv N \pmod{q_0}$. If t ranges along the arithmetic progression $t = uq_0 + r_0$, then

$$t^2 - N = (uq_0 + r_0)^2 - N,$$

where the right-hand side as a polynomial in u has coefficients divisible by q_0. Thus we have to add q_0 to the factorbase. This method of choosing special primes is sometimes referred to as special q method.

Montgomery came up with an improvement of this method, and this improvement has become known as the multiple polynomial quadratic sieve (cf. [45]). The polynomials are of the form $f(t) = At^2 + Bt + C \in \mathbb{Z}(t)$ with $B^2 - 4AC = N$. If we multiply $f(t)$ with $4A$, we get the polynomial

$$4Af(t) = (2At + B)^2 - (B^2 - 4AC) = (2At + B)^2 - N,$$

and this is a square mod N. If A is not smooth and not a square either, then we include A in the factorbase. By keeping only the F-smooth polynomial values, we are in the same situation as in the quadratic sieve. The only difference is that we switch to the next polynomial, when the polynomial values become too large. The downside of switching polynomials is that we have to compute the roots of each polynomial

modulo the primes of the factorbase. To overcome this, the self initializing quadratic sieve (SIQS) provides a fast way to switch polynomials. The main difference is that A is the product of a set of primes in the factorbase, $q_1 \dots q_s$. This leads to 2^{s-1} different polynomials and once the initialization of the first polynomial is done, the rest follows easily. For more details on QS, MPQS and SIQS, see, e.g., [32], [14], [43], [2] and [10].

An extra variation we should mention, as we use it a lot in this thesis, is the use of so-called large primes in both QS and MPQS. Besides the bound F, we choose an additional bound L, $L > F$. Instead of saving only polynomial values that are F-smooth, we also keep values that additionally have one or two primes between F and L, the so-called large primes [28]. Sometimes even three large primes are allowed [30]. If we have two relations, each with only one, and the same, large prime, we combine these two relations into one F-smooth relation (the large prime occurs twice, and forms a square). To reduce relations with two large primes to F-smooth relations, it might be necessary to combine several relations. This can be expressed as finding cycles in a graph, where the vertices represent large primes and the edges between two vertices relations which contain both the corresponding primes. To be able to represent relations with only one large prime as an edge, the number 1 is taken as a vertex too [28].

On the one hand, allowing large primes in the relations leads to more relations to process, which will take more time. On the other hand, we combine these relations into F-smooth relations, so we stop the sieving sooner. Overall, MPQS with two large primes becomes much faster than MPQS with one large prime: Lenstra and Manasse report a speed-up factor of 2 to 2.5 for N having more than 90 decimal digits [28]. Leyland et al. report that for their implementation and for integers of about 135 decimal digits using three primes is almost twice as effective as using only two [30]. However, for integers of this size the number field sieve will be much faster.

1.3 Number field sieve

The number field sieve (NFS) is also based on the idea of sieving and finding dependencies between the relations. However, the polynomials are quite different from the polynomials in MPQS. We describe all four steps of the number field sieve. First we select two irreducible polynomials $f_1(x)$ and $f_2(x)$, $f_1, f_2 \in \mathbb{Z}[x]$, and an integer $0 \leq m < N$, such that

$$f_1(m) \equiv f_2(m) \equiv 0 \pmod{N}.$$

Polynomials with 'small' integer coefficients are preferred, because the values of these polynomials are smaller on average and therefore more likely to be smooth than the values of polynomials with large integer coefficients. There are more criteria for finding good polynomials, such as the number of roots modulo small primes. We refer to the thesis of Murphy for the details [35]. Usually $f_1(x)$ is a (monic) linear polynomial and $f_2(x)$ a higher degree polynomial, referred to as rational side and algebraic side, respectively. The choice of a linear polynomial leaves more freedom for choosing a

suitable non-linear polynomial, but the (maximum) norms of the two polynomials are unbalanced. Kleinjung has developed a method for choosing non-monic linear polynomials, which usually leads to a better polynomial pair [23]. Montgomery has proposed a method to select two quadratic polynomials (cf. Section 4.5 of [18]), but it is an open problem to quickly construct two (or more) polynomials of higher degrees with a common root and relatively small coefficients. If N is of a special form (e.g., $c^n \pm 1$, denoted by $c, n\pm$) then we can use this to get a polynomial $f_2(x)$ with very small coefficients and $f_1(x)$ will be a linear polynomial (with the same root). In that case one speaks of the special number field sieve (SNFS), otherwise of the general number field sieve (GNFS). By α_1 and α_2 we denote roots of $f_1(x)$ and $f_2(x)$, respectively.

The second step is to collect relations. We choose a factorbase FB of all primes below the bound F, and a large primes bound L; for ease of exposition we take the same bounds on both the rational side and the algebraic side. Then we search for pairs (a, b) such that $\gcd(a, b) = 1$, and that both $F_1(a, b) := b^{\deg(f_1)} f_1(a/b)$ and $F_2(a, b) := b^{\deg(f_2)} f_2(a/b)$ have all their prime factors below F except for at most two prime factors, each between F and L, the so-called large primes. These pairs (a, b) are referred to below as relations. There are many options for collecting the relations, at present the fastest and most practical of which are based on sieving, for N in the range of 100–220 digits. Two sieving methods are widely used, viz. line sieving and lattice sieving. For line sieving we select a rectangular sieving area of points (a, b) and the sieving is done per horizontal line. For lattice sieving we select an interval of so-called special primes and for each special prime we only sieve those pairs (a, b) for which this special prime divides $F_2(a, b)$; for each special prime these pairs form a lattice in the sieving area, except when f_2 has multiple roots mod q. In case of SNFS the special primes are chosen on the rational side.

The third step uses linear algebra to construct squares on both the rational side and the algebraic side. As $F_1(a, b)$ is the norm of the algebraic number $a - b\alpha_1$, multiplied by the leading coefficient of $f_1(x)$, we first concentrate on the norm of the homogeneous polynomial to form a square. Theoretically, we know that the principal ideal $(a - b\alpha_1)$ factors into the product of prime ideals in the number field $\mathbb{Q}(\alpha_1)$. Only a few different prime ideals can appear in these factorizations, as all prime ideals appearing in these factorizations have norms at most L. Now use linear algebra to construct a set S of indices i such that the product $\prod_{i\in S}(a_i - b_i\alpha_1)$ is a square of products of prime ideals. If f_1 is a linear polynomial, we work with primes instead of prime ideals. The situation is similar for f_2. However, both sides have to be squares at the same time, so we look for a set S' of indices i such that the principal ideal products $\prod_{i\in S'}(a_i - b_i\alpha_1)$ and $\prod_{i\in S'}(a_i - b_i\alpha_2)$ are squares of products of prime ideals.

The last step is the square root step, of which we only give the main idea. We determine algebraic numbers $\alpha_1' \in \mathbb{Q}(\alpha_1)$ and $\alpha_2' \in \mathbb{Q}(\alpha_2)$ such that $(\alpha_1')^2 = \prod_{i\in S'}(a_i - b_i\alpha_1)$ and $(\alpha_2')^2 = \prod_{i\in S'}(a_i - b_i\alpha_2)$. Then we use the ring homomorphisms $\phi_{\alpha_1} : \mathbb{Z}[\alpha_1] \to \mathbb{Z}/N\mathbb{Z}$ and $\phi_{\alpha_2} : \mathbb{Z}[\alpha_2] \to \mathbb{Z}/N\mathbb{Z}$ with $\phi_{\alpha_1}(\alpha_1) = \phi_{\alpha_2}(\alpha_2) = m \bmod N$ to get $\phi_{\alpha_1}(\alpha_1')^2 = \phi_{\alpha_1}\left((\alpha_1')^2\right) = \phi_{\alpha_1}\left(\prod_{i\in S'}(a_i - b_i\alpha_1)\right) \equiv \prod_{i\in S'}((a_i - b_i m) \equiv \phi_{\alpha_2}(\alpha_2')^2 (\bmod N)$. Now we compute $\gcd(\phi_{\alpha_1}(\alpha_1') - \phi_{\alpha_2}(\alpha_2'), N)$ to obtain a factor of

N. If this gives a trivial factor (1 or N), then we proceed with another set of indices as above, otherwise we have found a nontrivial factorization of N. For more details on the NFS, see e.g., [18], [27], or [32].

1.4 Smooth and semismooth numbers

Smooth and semismooth numbers play an important role in factoring algorithms, such as MPQS and NFS. We let $n = n_1 n_2 \cdots$ with the prime factors $n_1, n_2, \ldots$ of n in nonincreasing order. Given a positive real number F, n is called F-smooth if $n_1 \leq F$. By $\Psi(x, F)$ we denote the number of positive integers $\leq x$ that are F-smooth, i.e.,

$$\Psi(x, F) = \#\{n \leq x : n_1 \leq F\}.$$

Given positive real numbers F and L, a number n is said to be k-semismooth with respect to F and L if n has exactly k prime factors between F and L, and all other prime factors $\leq F$. In factoring algorithms, F is much larger than $\log^2 x$. This implies that asymptotic approximating formulas for $\Psi(x, F)$ in the literature can be expressed in the so-called Dickman's ρ function, as in the works of de Bruijn and Ramaswami [8, 40, 41].

In Chapter 2 we derive a general asymptotic approximation for the number of smooth and k-semismooth numbers ($k = 1$ and $k = 2$ are treated separately). These approximations generalize the existing approximations of, e.g., Bach and Peralta, Cavallar, Knuth and Trabb Pardo, and Lambert [4, 9, 24, 25]. Our approximations consist of a main term, a second order term and an error term. The most general approximation, for k-semismooth numbers with (possibly) a different upper bound for each large prime is given in Theorem 7. In [47], Tenenbaum has given a general asymptotic expansion for the number of k-semismooth numbers with even higher (than second) order terms, in which the coefficients are given in an implicit way. Our theorems give explicit expressions for the coefficients in terms of integrals over functions depending on the Dickman ρ function.

As the results of Chapter 2 are asymptotic, it is not clear how good the main term is, compared with the number of (semi)smooth numbers found during the factorization of a given number. By comparing theoretical and practical results in Chapter 3, we study how good the theoretical predictions are and if they can be used to minimize the sieving time. Additionally, we look at the influence of the second order term. It turns out that there is often a considerable discrepancy between the theoretical estimates and the practical number of found (semi)smooth numbers, and that the second order term has a significant influence of about 10 % for numbers of approximately 100 decimal digits. Another aspect we look at is the use of different upper bounds for the large primes, instead of using equal bounds. We study how different choices of the upper bounds for the large primes affect the final result, when we keep the product of the bounds constant.

1.5 Simulating the sieving

The asymptotic formula $L(N)$ for the running time of the number field sieve (cf. Section 1.1) is not useful in practice to predict the sieving time. If somebody is experienced with factoring numbers with NFS, (s)he could use this experience, but still the prediction can easily be 10 % off. This is due to the growth behavior of the number of relations after singleton removal, where the difficulty is the unpredictable matching behavior of the primes in the relations, as the primes of a newly found relation may or may not match with the primes of earlier found relations.

In Chapter 4 we propose a method for predicting the number of necessary relations for factoring a given N with NFS with given parameters, and the corresponding sieving time. The basic idea is to do a small but representative amount of sieving and analyze the relations in this sample. We randomly generate pairs of rational and algebraic large primes combinations according to the relevant distributions as observed in the sample, and hope that the matching behavior of these cheaply generated *simulated relations* corresponds to the matching behavior of the actual relations. Thus, during the simulation we regularly count the number of simulated relations after singleton removal and assume that the point where the number of simulated relations would generate an oversquare matrix is a good estimate for the number of actual relations that will be required. Experiments show that our predictions of the number of necessary relations are within 2 % of the number of relations needed in the real factorization (cf. Section 4.4).

In Section 4.2 we give the details of simulating relations for Case I and Case II; if F and L are relatively close we consider it as Case I, else we consider it as Case II. The simulation is valid for both line sieving and lattice sieving. In Section 4.3 we go into the details of the stop criterion. After giving the results of our experiments in Section 4.4, we present in Section 4.5 a way to determine in advance if Case I or Case II applies.

We conclude the chapter with additional experiments on certain details of our prediction method. We found experimentally that a sieving test of about 0.1 % of the relations already gives good results. By using Chebyshev's inequality, we are able to compute the appropriate size (or percentage of sieving points) of the sieving test (cf. Subsection 4.6.1). Related to determining the size of the sieving test is the determination of the total sieving area as this influences the amount of possible relations and the sieving time. We explain in Subsection 4.6.2 how to get a fitting sieving area and a good estimate of the total sieving time. In Subsection 4.6.3 we concentrate on the growth behavior of the relations after singleton removal. It is described as explosive by Dodson and Lenstra [17], but a more gradual growth does also occur. This seems to depend on the sieving bounds that are chosen. In Subsection 4.6.4 we study the effect of the number of (simulated) relations on the size of the resulting matrix.

1.6 Conclusion

In Chapter 5 we summarize the results as given in this thesis. Furthermore, we give suggestions for further research, related to the subject.

Chapter 2

Smooth and Semismooth Numbers

2.1 Introduction

Smooth numbers are positive integers without large prime factors. They have been studied extensively and the results play an important role in several number theoretical topics, e.g., large gaps between primes [42] and the number field sieve [27]. These numbers are also used in numerical problems, such as the Cooley-Tukey FFT algorithm [11]. The first results go back to the 1920's, when Vinogradov reported on a bound for the least nth power non-residue [48].

In the present chapter we give an approximating function for k-semismooth numbers with (possibly) a different upper bound for each large prime. This approximating function consists of a main term, a second order term and an error term, where all terms are given explicitly (see Theorem 7). For use in Chapter 3 we explicitly state the results for 1- and 2-semismooth numbers as well. Our main motivation for looking at theoretical approximations of the number of (semi)smooth numbers is that factoring algorithms such as MPQS and NFS search for (semi)smooth numbers during the sieving step. By using good asymptotic approximation expansions with a second order term, we may be able to improve upon the overall running time of factoring algorithms in practice. In Chapter 3 we compare the theoretical approximations of smooth, 1-semismooth, and 2-semismooth numbers with the number of (semi)smooth numbers found during the sieving step of MPQS and NFS.

Recall that a y-smooth number is a positive integer with no prime factors larger than y. We write $n = n_1 n_2 \cdots$ with n_i prime and $n_1 \geq n_2 \geq \ldots$, and $y = x^\alpha$, where $0 < \alpha < 1$. Then the number of positive integers $\leq x$ that are y-smooth can be written as

$$\Psi(x, x^\alpha) = \#\{n \leq x : n_1 \leq x^\alpha\}.$$

To compute this number, one could use a sieve procedure to find which numbers are y-smooth. This is a lot of work when x is large. However, in many situations it suffices to have a good estimate. Approximating functions of the Ψ-function have been found with the help of asymptotic expansions. Ramaswami in 1948 [40, 41] and De Bruijn in 1951 [8] gave approximations of the Ψ-function, which we display in the next sections.

In 1970 Norton [36] and in 1993 Hildebrand and Tenenbaum [22] wrote overviews of what was known at that time. More recent work on the Ψ-function by Parsell and Sorensen concentrates on finding rigorous upper and lower bounds for this function [38]. Bernstein worked as well on finding tight bounds on the distribution of smooth integers [5]. Furthermore Suzuki [46] has given a fast algorithm for approximating $\Psi(x, y)$, based on an algorithm of Hildebrand and Tenenbaum.

As most numbers kept during the sieving step of factoring algorithms are semismooth, we need to study the corresponding Ψ-functions. To be more specific, a 1-semismooth number is a smooth number with all its prime factors at most x^α, with the exception of one prime factor $> x^\alpha$, which does not exceed a larger bound x^β, with $0 < \alpha < \beta < 1$. The analogue of the Ψ-function for 1-semismooth numbers is defined as

$$\Psi_1(x, x^\beta, x^\alpha) = \#\{n \leq x : n_2 \leq x^\alpha < n_1 \leq x^\beta\}.$$

A 2-semismooth number has two prime factors exceeding a bound x^α, but not exceeding a bound x^β. These prime factors are often referred to as large primes. For the upper bound on the two large primes we have two choices: an equal bound for both primes or non-equal bounds. As far as we know, choosing different upper bounds for the large primes is a new aspect in the analysis of semismooth numbers. If we choose the same bound x^β for both large primes, the definition of the corresponding Ψ-function is

$$\Psi_2(x, x^\beta, x^\alpha) = \#\{n \leq x : n_3 \leq x^\alpha < n_2 \leq n_1 \leq x^\beta\}.$$

If we choose different upper bounds for the two large primes, the definition of the corresponding Ψ-function is

$$\Psi_2(x, x^{\beta_1}, x^{\beta_2}, x^\alpha) = \#\{n \leq x : n_3 \leq x^\alpha < n_2 \leq n_1 \leq x^{\beta_1}, n_2 \leq x^{\beta_2}\},$$

with $0 < \alpha < \beta_2 \leq \beta_1 < 1$. If $\beta_2 = \beta_1$, we have again the same upper bound for both large primes.

This generalizes to k-semismooth numbers. A k-semismooth number is a number with all its prime factors below a certain bound x^α, except for k prime factors between x^α and x^β, with $\alpha < \beta < 1$. The definition for k-semismooth numbers with the same upper bound for the large primes can be written as

$$\Psi_k(x, x^\beta, x^\alpha) = \#\{n \leq x : n_{k+1} \leq x^\alpha < n_k, n_1 \leq x^\beta\}.$$

We consider different bounds for the large prime factors as well. Let α and $\beta_1, \ldots, \beta_k$ be numbers with $0 < \alpha < \beta_k \leq \ldots \leq \beta_1 < 1$. We put

$$\Psi_k(x, x^{\beta_1}, \dots, x^{\beta_k}, x^{\alpha}) =$$

$$\#\{n \le x : n_{k+1} \le x^{\alpha}, x^{\alpha} < n_k \le x^{\beta_k}, n_k \le n_{k-1} \le x^{\beta_{k-1}}, \dots, n_2 \le n_1 \le x^{\beta_1}\}.$$

In Section 2.2 we give some tools that we frequently use in the proofs. In Sections 2.3 and 2.4 we give known and new approximating functions for the different Ψ_i-functions. These asymptotic expansions consist of a main term, zero or more higher order terms, and an error term. We denote asymptotic expansions with only a main term and an error term by Type 1 and those with a main term, higher order terms and an error term by Type 2. In Section 2.3 we give the asymptotic expansions of Type 1 for smooth and semismooth numbers, and the Type 2 expansions are put together in Section 2.4. In Section 2.5 we give the proof of Theorem 6, which is an improvement of a result of Ramaswami. Finally, in Section 2.6 we give the proof of Theorem 7, which is our Type 2 asymptotic expansions for k-semismooth numbers with a different upper bound for each large prime.

2.2 Basic tools

Let $\pi(x)$ denote the number of primes $\le x$. In this paper we will often use the Prime Number Theorem, which says that

$$\pi(x) = \mathrm{li}(x) + \epsilon(x), \tag{2.1}$$

with $\mathrm{li}(x) = \int_0^x \mathrm{d}t/\log t$ and $\epsilon(x) = o(x/\log x)$ for $x \to \infty$. De la Vallée Poussin proved that one may take $\epsilon(x) = O(x\mathrm{e}^{-C\sqrt{\log x}})$, where C is a positive constant. It follows that $\epsilon(x) = O(x/\log^c x)$ for any $c > 1$. Using Stieltjes integration and (2.1) leads to

$$\sum_{p \text{ prime}, p<x} \frac{1}{p} = \log\log x + O(1), \text{ as } x \to \infty, \tag{2.2}$$

and

$$\sum_{p \text{ prime}, p<x} \frac{1}{p\log p} = O(1), \text{ as } x \to \infty; \tag{2.3}$$

see for more details, e.g., [20]. We shall often tacitly assume that $x \to \infty$ when we use the O- and o-term notations.

2.3 Type 1 expansions

We give here asymptotic expansions that consist only of a main term and an error term. To keep an overview of the historical developments the results on smooth numbers are given in Subsection 2.3.1, on 1-semismooth numbers in Subsection 2.3.2, on 2-semismooth numbers in Subsection 2.3.3 and on k-semismooth numbers for general k in Subsection 2.3.4.

2.3.1 Smooth numbers

The main result on smooth numbers goes back to a result of Dickman, who improved Vinogradov's estimate for $\Psi(x, x^\alpha)$. Recall that we write the number of positive integers $\leq x$ that are y-smooth as $\Psi(x, x^\alpha) = \#\{n \leq x : n_1 \leq x^\alpha\}$. Dickman showed that $\Psi(x, x^\alpha) \sim x\rho(1/\alpha)$ $(x \to \infty)$ for each fixed $\alpha > 0$. The ρ function is the so-called Dickman ρ function, which is the unique continuous solution of the differential-difference equation

$$\begin{cases} \rho(x) = 1 & 0 \leq x \leq 1 \\ \rho'(x) = -\rho(x-1)/x & x > 1. \end{cases}$$

De Bruijn improved the result by giving a range of α for which the approximation of Dickman is valid uniformly in α. Hildebrand [21, 22] provided De Bruijn's asymptotic value for $\Psi(x, x^\alpha)$ [8] with a uniform error term, i.e. the dependence on the used parameter α is explicitly given. The result is given in the following theorem.

Theorem 1 (Hildebrand) *For any fixed $\epsilon > 0$ the relation*

$$\Psi(x, x^\alpha) = x\rho\left(\frac{1}{\alpha}\right)\left(1 + O\left(\frac{\log(1/\alpha)}{\alpha \log x}\right)\right), \text{ as } x \to \infty,$$

holds uniformly in the range $x^\alpha \geq 2$, $1 \leq \frac{1}{\alpha} \leq \exp\left((\alpha \log x)^{3/5-\epsilon}\right)$.

We illustrate the range, as given in Theorem 1, in Figure 2.1 for $\epsilon = 0$, where the area marked with diagonal lines is the range mentioned in the theorem. We use $X = \log x$ and $Y = \alpha \log x$ as X- and Y-axis, respectively.

2.3.2 1-semismooth numbers

One of the first approximations of $\Psi_1(x, x^\beta, x^\alpha)$ is due to Bach and Peralta [4]. We state their result as follows.

Theorem 2 (Bach and Peralta) *If $0 < \alpha < \beta < 1$ and $x^\alpha \geq 2$, then*

$$\Psi_1(x, x^\beta, x^\alpha) = x \int_\alpha^\beta \rho\left(\frac{1-\lambda}{\alpha}\right)\frac{\mathrm{d}\,\lambda}{\lambda} + O\left(\frac{\log(1/\alpha)}{\alpha(1-\beta)}\frac{x}{\log x}\right).$$

Compared with Theorem 3.1 in [4], where only the condition $0 < \alpha < \beta < 1$ is stated, we have added the condition $x^\alpha \geq 2$. However, this extra condition should be imposed in Theorem 3.1 of Bach and Peralta as well. It is a consequence of the necessary addition of the condition $t^\gamma \geq 2$ in Formula 2.7 of [4].

Lemma 1 (cf. [4], (2.7)) *Results of De Bruijn imply that if $0 < \gamma < 1$ and $t^\gamma \geq 2$, we have*

$$\Psi(t, t^\gamma) = t\rho(1/\gamma) + O(\frac{t}{\gamma \log t}).$$

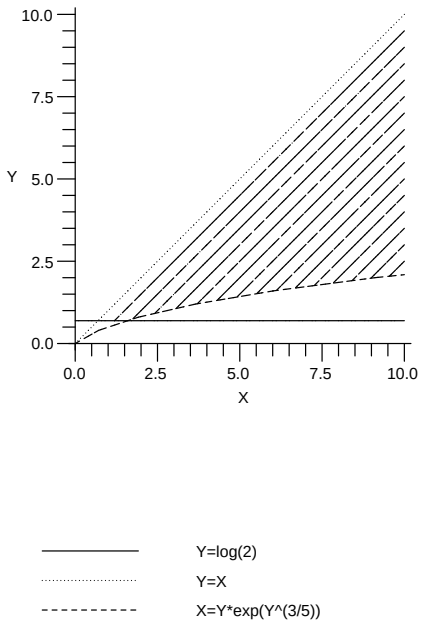

Figure 2.1: Range of $X = \log x$ and $Y = \alpha \log x$ for which Theorem 1 holds

To prove this, combine (5.3) and (5.4) of [8], instead of (1.4) and (5.3). With the latter combination one can prove Lemma 1 as well, but only subject to extra conditions. Bach and Peralta apply Lemma 1 for $t = \frac{x}{p}$ and $\gamma = \frac{\alpha}{1-\log p/\log x}$ with $x^\alpha < p \le x^\beta$. Thus $t^\gamma \ge 2$ leads to $x^\alpha \ge 2$.

The main term and O-term in Formula 3.8 of [4] remain the same, as we know that $\alpha \le \frac{\alpha}{1-\log p/\log x}$. Two minor corrections in the proof concern the O-term in the third displayed equation on page 1705 of [4] and the fifth displayed equation on the same page. The O-term should be $O\left(\frac{1}{\alpha \log x}\right)$ and in the fifth equation there should be an x in front of the integral.

More details of the proof of Theorem 2 can be found in [4]. A more precise error term will be given in Corollary 2 for $k = 1$.

2.3.3 2-semismooth numbers

For semismooth numbers with two large primes with the same upper bound, Lambert ([25], Ch. 4) used the result of Bach and Peralta to obtain

$$\Psi_2(x, x^\beta, x^\alpha) = \frac{x}{2} \int_\alpha^\beta \int_\alpha^\beta \rho\left(\frac{1-\lambda_1-\lambda_2}{\alpha}\right) \frac{\mathrm{d}\lambda_1}{\lambda_1} \frac{\mathrm{d}\lambda_2}{\lambda_2} +$$
$$O\left(\frac{\log(\beta/\alpha)x}{\alpha \log x}\left(\log\left(\frac{\beta(1-2\alpha)}{\alpha(1-2\beta)}\right)+1\right)\right), x \to \infty. \tag{2.4}$$

The factor $\frac{1}{2}$ in front of the integral is to correct for the integration area, which is $\alpha < \lambda_1 < \beta$ and $\alpha < \lambda_2 < \beta$. However, we impose $\lambda_2 \leq \lambda_1$, hence a factor $\frac{1}{2}$ (as the integrand is symmetric in λ_1 and λ_2).

We now give the result for 2-semismooth numbers with different upper bounds on the large primes.

Theorem 3 *Let $\epsilon > 0$ be fixed. If $0 < \alpha < \beta_2 < \beta_1$, $\alpha + \beta_2 + \beta_1 \leq 1$, $x^\alpha \geq 2$, and $\frac{1-2\alpha}{\alpha} \leq \exp((\frac{\alpha}{1-2\alpha}\log x)^{3/5-\epsilon})$, then we have for $x \to \infty$,*

$$\Psi_2(x, x^{\beta_1}, x^{\beta_2}, x^\alpha) =$$
$$x \int_\alpha^{\beta_2} \int_{\lambda_2}^{\beta_1} \rho\left(\frac{1-\lambda_1-\lambda_2}{\alpha}\right) \frac{\mathrm{d}\lambda_1}{\lambda_1} \frac{\mathrm{d}\lambda_2}{\lambda_2} \left(1 + O\left(\frac{\log(\frac{1}{\alpha})}{\alpha \log x}\right)\right).$$

The proof is based on Theorem 1, and the structure of the proof is similar to the proof in Section 2.6. As a corollary we give the result for equal upper bounds on the large primes. The result is a refinement of Lambert's result. The main difference is the error term, which now includes the ρ function, which decreases exponentially fast.

Corollary 1 *Let $\epsilon > 0$ be fixed. If $0 < \alpha < \beta$, $\alpha + 2\beta \leq 1$, $x^\alpha \geq 2$, and $\frac{1-2\alpha}{\alpha} \leq \exp((\frac{\alpha}{1-2\alpha}\log x)^{3/5-\epsilon})$, then, for $x \to \infty$,*

$$\Psi_2(x, x^\beta, x^\alpha) = \frac{x}{2} \int_\alpha^\beta \int_\alpha^\beta \rho\left(\frac{1-\lambda_1-\lambda_2}{\alpha}\right) \frac{\mathrm{d}\lambda_1}{\lambda_1} \frac{\mathrm{d}\lambda_2}{\lambda_2} \left(1 + O\left(\frac{\log(\frac{1}{\alpha})}{\alpha \log x}\right)\right).$$

2.3.4 k-semismooth numbers

One of the known results for k-semismooth numbers with equal bounds on the large primes can be found in the thesis of Cavallar [9]. The result is based on the results of Bach and Peralta and of Lambert. Cavallar's result is the following. Note that all integrals have bounds α and β. This is compensated by a factor $1/k!$.

For a positive integer k, $0 < \alpha < \beta < 1/k$ and $\log x > \frac{1}{\alpha}\max(\log 2, \frac{1-k\alpha}{\alpha}, \frac{1}{\log((k\alpha)^{-1})})$ we have

$$\Psi_k(x, x^\beta, x^\alpha) = \frac{x}{k!} \int_\alpha^\beta \cdots \int_\alpha^\beta \rho\left(\frac{1-(\lambda_1+\cdots\lambda_k)}{\alpha}\right) \frac{\mathrm{d}\lambda_1}{\lambda_1} \cdots \frac{\mathrm{d}\lambda_k}{\lambda_k} +$$
$$O\left(\frac{\log^k((k\alpha)^{-1})}{\alpha(1-k\beta)} \frac{x}{\log x}\right).$$

The error bound is uniform in k, α and β.

Our result for k-semismooth numbers with different upper bounds on the large prime factors is the following theorem.

Theorem 4 *Let $\epsilon > 0$ be fixed. If $0 < \alpha < \beta_k \leq \ldots \leq \beta_1$, $\alpha + \beta_k + \ldots + \beta_1 \leq 1$, $x^\alpha \geq 2$, and $\frac{1-k\alpha}{\alpha} \leq \exp((\frac{\alpha}{1-k\alpha}\log x)^{3/5-\epsilon})$, then we have for $x \to \infty$,*

$$\Psi_k(x, x^{\beta_1}, \ldots, x^{\beta_k}, x^\alpha) =$$

$$x\int_\alpha^{\beta_k}\int_{\lambda_k}^{\beta_{k-1}}\cdots\int_{\lambda_2}^{\beta_1}\rho\left(\frac{1-(\lambda_1+\ldots+\lambda_k)}{\alpha}\right)\frac{\mathrm{d}\lambda_1}{\lambda_1}\cdots\frac{\mathrm{d}\lambda_\mathrm{k}}{\lambda_k}\left(1+O\left(\frac{\log(\frac{1}{\alpha})}{\alpha\log x}\right)\right).$$

The proof is based on Theorem 1 as well and consists of the same basic ideas as the proof in Section 2.6. In case of equal bounds on the large prime factors, we have the following corollary.

Corollary 2 *For any fixed $\epsilon > 0$, $k \geq 1$ ($k \in \mathbb{Z}_+$), if $0 < \alpha < \beta$, $x^\alpha \geq 2$, $\alpha + k\beta \leq 1$, and $\frac{1-k\alpha}{\alpha} \leq \exp((\frac{\alpha}{1-k\alpha}\log x)^{3/5-\epsilon})$, then we have for $x \to \infty$,*

$$\Psi_k(x, x^\beta, x^\alpha) =$$

$$\frac{x}{k!}\int_\alpha^{\beta}\cdots\int_\alpha^{\beta}\rho\left(\frac{1-(\lambda_1+\lambda_2+\ldots+\lambda_k)}{\alpha}\right)\frac{\mathrm{d}\lambda_1}{\lambda_1}\cdots\frac{\mathrm{d}\lambda_\mathrm{k}}{\lambda_k}\left(1+O\left(\frac{\log(\frac{1}{\alpha})}{\alpha\log x}\right)\right).$$

The most important difference with Cavallar's result is that our O-term involves the ρ function, which decreases exponentially fast.

2.4 Type 2 expansions

In this section we give asymptotic expansions that consist of a main term, a second order term and an error term. We start with results on smooth numbers in Subsection 2.4.1, and continue with results on k-semismooth numbers in Subsection 2.4.2. We also give our results for $k = 1$ and $k = 2$ as corollaries of Theorem 7.

2.4.1 Smooth numbers

Ramaswami [41] gave an approximation of the Ψ-function by adding a second order term as in the following theorem. See also Norton [36], p. 12.

Theorem 5 (Ramaswami) *For $x > 1$, $0 < \alpha < 1$, and $x^\alpha > 2$ we have*

$$\Psi(x, x^\alpha) = x\rho\left(\frac{1}{\alpha}\right) + (1-\gamma)\frac{x}{\log x}\rho\left(\frac{1-\alpha}{\alpha}\right) + O_\alpha(\Delta(x, x^\alpha)), \text{ as } x \to \infty,$$

where

$$\Delta(x, x^\alpha) = \begin{cases} \frac{x}{(\log x)^{3/2}} & \text{for } 0 < \alpha < 1/2, \\ \frac{x^\alpha}{\log x} + \frac{x}{\log^2 x} & \text{for } 1/2 \leq \alpha < 1. \end{cases}$$

In this theorem γ is Euler's constant. We now improve the error term and make the dependence on α explicit. Knuth and Trabb Pardo [24] proved that

$$\Psi_k(x, x^\alpha) = x\rho_k(\frac{1}{\alpha}) + \frac{x}{\log x}\sigma_k(\frac{1}{\alpha}) + O_\alpha\left(\frac{x}{\log^2 x}\right),$$

as $x \to \infty$, for all fixed $0 < \alpha < 1$. Here, $\Psi_k(x, x^\alpha)$ stands for the number of positive integers up to x with their kth largest prime factor below x^α, and $\sigma_k(\frac{1}{\alpha})$ is defined as $(1-\gamma)(\rho_k(\frac{1}{\alpha}-1) - \rho_{k-1}(\frac{1}{\alpha}-1))$, where

$$\rho_k(\frac{1}{\alpha}) = 1 - \int_1^{\frac{1}{\alpha}} (\rho_k(t-1) - \rho_{k-1}(t-1))\,\frac{\mathrm{d}t}{t}, \text{ for } 0 < \alpha < 1, k \geq 1,$$

$$\rho_k(\frac{1}{\alpha}) = 1 \text{ for } \alpha \geq 1, k \geq 1, \text{ and}$$

$$\rho_k(\frac{1}{\alpha}) = 0 \text{ for } \alpha < 0 \text{ or } k = 0.$$

Using their approach we improve the error term of Theorem 5 as follows.

Theorem 6 *For $0 < \alpha < 1$ and $x^\alpha > 2$ we have*

$$\Psi(x, x^\alpha) = x\rho\left(\frac{1}{\alpha}\right) + (1-\gamma)\frac{x}{\log x}\rho\left(\frac{1-\alpha}{\alpha}\right) + O\left(\frac{x}{\alpha^3\log^2 x}\right), \text{ as } x \to \infty.$$

In order to keep the overview on the various results on (semi)smooth numbers, we give the proof of Theorem 6 in Section 2.5. In the sequel of this section we extend the Ψ-function for smooth numbers to semismooth numbers and generalize Theorem 6 to such numbers.

2.4.2 k-semismooth numbers

For k-semismooth numbers there exist some results in case of equal bounds on the large primes. We start with the result of Zhang [50], who made use of recursively defined functions. To be precise, Zhang states that for k (> 0) large primes between x^α and x^β, and $0 < \alpha < \beta < 1$

$$\Psi_k(x, x^\beta, x^\alpha) = xG_k(\alpha, \beta) + \frac{x}{\log x}\lambda_k(\alpha, \beta) + O_\alpha\left(\frac{x}{\log^2 x}\right), \tag{2.5}$$

where

$$G_k(\alpha, \beta) = F(\alpha) + \int_\alpha^\beta G_{k-1}\left(\frac{\alpha}{1-t}, \frac{t}{1-t}\right)\frac{\mathrm{d}t}{t},$$

with $F(\alpha) = G_0(\alpha, \beta) = \rho(1/\alpha)$ and

$$\lambda_k(\alpha, \beta) = (1-\gamma)F\left(\frac{\alpha}{1-\alpha}\right) + \int_\alpha^\beta \lambda_{k-1}\left(\frac{\alpha}{1-t}, \frac{t}{1-t}\right)\frac{\mathrm{d}t}{t(1-t)},$$

with $\lambda_0(\alpha,\beta) = (1-\gamma)F(\alpha/(1-\alpha))$.

Another result on k-semismooth numbers is due to Tenenbaum [47]. Tenenbaum defines $m_1, m_2, ..., m_k$ as the decreasing sequence of distinct prime factors of a positive integer m. (Note that Tenenbaum does not take multiplicities of the prime factors into account.) Then it is possible to provide an asymptotic expansion for the distribution function $F_n(\vec{\alpha_k}) := \nu_n\{m : m_j > n^{\alpha_j}(1 \le j \le k)\}$, which is valid uniformly in a large range for $\vec{\alpha_k} := (\alpha_1, \ldots, \alpha_k)$. The main theorem of [47] consists of four items; for our purpose the third item is the most interesting:

Let k be a positive integer. There exists a sequence of real functions $\{\phi_h\}_{h=0}^{\infty}$ defined on $[0,1]^k$, an increasing sequence of integers $\{R_h\}_{h=0}^{\infty}$ with $R_0 = 0$, and a sequence of affine linear forms $\{\Delta_r(\vec{\alpha_k})\}_{r=1}^{\infty}$ having the following property:
For arbitrary but fixed $H \ge 0$ and $\varepsilon \in (0, 1/3)$, we have

$$F_n(\vec{\alpha_k}) = \sum_{0\le h\le H} \frac{\phi_h(\vec{\alpha_k})}{(\log n)^h} + O_{H,\varepsilon}\left(\frac{1}{(\alpha_k \log n)^{H+1}}\right),$$

uniformly in the range

$$\begin{cases} \vec{\alpha_k} \in [0,1]^k, \alpha_k > \kappa(\varepsilon, n), \\ \min_{1\le r\le R_h, \Delta_r(\vec{\alpha_k})>0} \Delta_r(\vec{\alpha_k}) > K_H \frac{\log_2 n}{\log n}, \end{cases}$$

where K_H is a suitable constant depending only on H.

This result contains even higher order terms, but the terms do not display any explicit dependency on the bounds of the large primes. By using the law of inclusion and exclusion the result can be transformed into a formula of the form (2.5).

We introduce an explicit second order term and error term on k-semismooth numbers with different bounds on the large primes in the following theorem.

Theorem 7 *If $0 < \alpha < \beta_k \le \ldots \le \beta_1$, $\alpha + \beta_k + \ldots + \beta_1 \le 1$, and $x^\alpha \ge 2$, then we have for $x \to \infty$*

$$\Psi_k(x, x^{\beta_1}, \ldots, x^{\beta_k}, x^\alpha) =$$
$$x \int_\alpha^{\beta_k} \int_{\lambda_k}^{\beta_{k-1}} \cdots \int_{\lambda_2}^{\beta_1} \rho\left(\frac{1-(\lambda_1+\ldots+\lambda_k)}{\alpha}\right) \frac{\mathrm{d}\lambda_1}{\lambda_1} \cdots \frac{\mathrm{d}\lambda_k}{\lambda_k} +$$
$$(1-\gamma)\frac{x}{\log x} \int_\alpha^{\beta_k} \int_{\lambda_k}^{\beta_{k-1}} \cdots \int_{\lambda_2}^{\beta_1} \rho\left(\frac{1-(\lambda_1+\ldots+\lambda_k)-\alpha}{\alpha}\right) \times$$
$$\frac{1}{1-(\lambda_1+\ldots+\lambda_k)} \frac{\mathrm{d}\lambda_1}{\lambda_1} \cdots \frac{\mathrm{d}\lambda_k}{\lambda_k} + O\left(\frac{\log(\beta_1/\alpha)\ldots\log(\beta_k/\alpha)}{\alpha^3(1-(\beta_1+\cdots+\beta_k))^2} \frac{x}{\log^2 x}\right).$$

We give the proof of this theorem in Section 2.6. In case of equal bounds on the large primes, we have the following corollary. This is comparable with Zhang's result (2.5), but the coefficients of the x- and $x/\log x$-terms and of the error term are more explicit than in (2.5).

Corollary 3 *If $0 < \alpha < \beta$, $\alpha + k\beta < 1$, and $x^\alpha \geq 2$, then we have for $x \to \infty$*

$$\Psi_k(x, x^\beta, x^\alpha) = \frac{x}{k!} \int_\alpha^\beta \int_\alpha^\beta \cdots \int_\alpha^\beta \rho\left(\frac{1-(\lambda_1+\ldots+\lambda_k)}{\alpha}\right) \frac{\mathrm{d}\lambda_1}{\lambda_1} \cdots \frac{\mathrm{d}\lambda_\mathrm{k}}{\lambda_k} +$$
$$\frac{1-\gamma}{k!} \frac{x}{\log x} \int_\alpha^\beta \int_\alpha^\beta \cdots \int_\alpha^\beta \rho\left(\frac{1-(\lambda_1+\ldots+\lambda_k)-\alpha}{\alpha}\right) \times$$
$$\frac{1}{1-(\lambda_1+\ldots+\lambda_k)} \frac{\mathrm{d}\lambda_1}{\lambda_1} \cdots \frac{\mathrm{d}\lambda_\mathrm{k}}{\lambda_k} + O\left(\frac{\log^k(\beta/\alpha)}{\alpha^3(1-k\beta)^2} \frac{x}{\log^2 x}\right).$$

In Chapter 3 we compare our theoretical results on (semi)smooth numbers with the amount of (semi)smooth numbers found during the sieving step of factoring algorithms. This concerns only 1- and 2- semismooth numbers, so we explicitly state Theorem 7 for $k = 1, 2$ in the following two corollaries.

Corollary 4 *If $0 < \alpha < \beta < 1$, $\alpha + \beta \leq 1$, and $x^\alpha \geq 2$, then*

$$\Psi_1(x, x^\beta, x^\alpha) = x \int_\alpha^\beta \rho\left(\frac{1-\lambda}{\alpha}\right) \frac{\mathrm{d}\lambda}{\lambda} +$$
$$(1-\gamma) \frac{x}{\log x} \int_\alpha^\beta \rho\left(\frac{1-\lambda-\alpha}{\alpha}\right) \frac{\mathrm{d}\lambda}{\lambda(1-\lambda)} + O\left(\frac{\log(\beta/\alpha)}{\alpha^3(1-\beta)^2} \frac{x}{\log^2 x}\right) \text{ for } x \to \infty.$$

Besides being a consequence of Theorem 7, this result can be seen as a generalization of the results of Ramaswami and of Bach and Peralta. For 2-semismooth numbers we only give the result on using different bounds on the two large primes. In factoring algorithms it might be useful to have different bounds for the large primes, as it might be used to improve the overall running time of the factoring algorithm.

Corollary 5 *If $0 < \alpha < \beta_2 < \beta_1$, $\alpha + \beta_2 + \beta_1 \leq 1$, and $x^\alpha \geq 2$, then we have for $x \to \infty$*

$$\Psi_2(x, x^{\beta_1}, x^{\beta_2}, x^\alpha) = x \int_\alpha^{\beta_2} \int_{\lambda_2}^{\beta_1} \rho\left(\frac{1-\lambda_1-\lambda_2}{\alpha}\right) \frac{\mathrm{d}\lambda_1}{\lambda_1} \frac{\mathrm{d}\lambda_2}{\lambda_2} +$$
$$(1-\gamma) \frac{x}{\log x} \int_\alpha^{\beta_2} \int_{\lambda_2}^{\beta_1} \rho\left(\frac{1-\lambda_1-\lambda_2-\alpha}{\alpha}\right) \frac{1}{1-\lambda_1-\lambda_2} \frac{\mathrm{d}\lambda_1}{\lambda_1} \frac{\mathrm{d}\lambda_2}{\lambda_2} +$$
$$O\left(\frac{\log(\beta_1/\alpha)\log(\beta_2/\alpha)}{\alpha^3(1-\beta_1-\beta_2)^2} \frac{x}{\log^2 x}\right).$$

2.5 Proof of Theorem 6

In this section we give the proof of Theorem 6, which is the following.
For $0 < \alpha < 1$ and $x^\alpha > 2$ we have

$$\Psi(x, x^\alpha) = x\rho\left(\frac{1}{\alpha}\right) + (1-\gamma)\frac{x}{\log x}\rho\left(\frac{1-\alpha}{\alpha}\right) + O\left(\frac{x}{\alpha^3 \log^2 x}\right), \text{ as } x \to \infty.$$

Proof. We define $S(x,y)$ as the set of y-smooth numbers at most x, thus we have $\Psi(x,y) = |S(x,y)|$. Furthermore, in summations we let p range over primes and n over positive integers.

We start the proof by expressing the difference between x and $\Psi(x,x^\alpha)$ in terms of the Ψ-function as

$$\begin{aligned}\lfloor x\rfloor - \Psi(x,x^\alpha) &= \sum_{x^\alpha<p\le x} \#\{n\le x : n_1 = p\}\\ &= \sum_{x^\alpha<p\le x} \#\left\{m\le \frac{x}{p} : m_1\le p\right\}\\ &= \sum_{x^\alpha<p\le x} \Psi\left(\frac{x}{p},p\right). \end{aligned} \tag{2.6}$$

The next step consists of replacing the sum by an integral. We have

$$V := \sum_{x^\alpha<p\le x} \Psi\left(\frac{x}{p},p\right) - \int_{x^\alpha}^{x} \Psi\left(\frac{x}{y},y\right)\frac{\mathrm{d}y}{\log y} =$$

$$= \sum_{x^\alpha<p\le x}\ \sum_{m\in S(x/p,p)} 1 - \int_{x^\alpha}^{x}\left(\sum_{m\in S(x/y,y)} 1\right)\frac{\mathrm{d}y}{\log y}.$$

Now use the definition of $S(x,y)$. If $m\in S(x/y,y)$, then $m\le x/y$ and $m_1\le y$. Combining this with the boundary condition of the integral and the combination of the two summations over m, we get

$$V = \sum_{\substack{1\le m\le x^{1-\alpha}\\ m_1\le x/m}}\left(\sum_{\substack{m_1\le p\le x/m\\ x^\alpha<p}} 1 - \int_{\max(m_1,x^\alpha)}^{x/m}\frac{\mathrm{d}y}{\log y}\right)$$

$$= \sum_{\substack{1\le m\le x^{1-\alpha}\\ m_1\le x/m}}\left(\pi\left(\frac{x}{m}\right) - \pi(\max(m_1,x^\alpha)) + O(1) - \mathrm{li}\left(\frac{x}{m}\right) + \mathrm{li}(\max(m_1,x^\alpha))\right).$$

By using De la Vallée Poussin's error term formula from Section 2.2, we obtain

$$V = \sum_{\substack{1\le m\le x^{1-\alpha}\\ m_1\le x/m}} O\left(\frac{x}{m}\mathrm{e}^{-C\sqrt{\log(x/m)}}\right) = \sum_{\substack{1\le m\le x^{1-\alpha}\\ m_1\le x/m}} O\left(\frac{x}{m}\mathrm{e}^{-C\sqrt{\alpha\log x}}\right).$$

Using $\sum_{m\le x^{1-\alpha}} 1/m = \log x^{1-\alpha} + \gamma + O(1/x^{1-\alpha})$, where γ is Euler's constant, leads to

$$V = O\left(x\log x\,\mathrm{e}^{-C\sqrt{\alpha\log x}}\right). \tag{2.7}$$

The O-term of (2.7) is $O\left(\frac{x\log x}{(\alpha\log x)^r}\right)$ for any positive number r, as $\mathrm{e}^{-\sqrt{\log x}} < \frac{1}{(\log x)^r}$ for $x \to \infty$, for any positive number r. We will use $r = 3$. Combining (2.6) and (2.7) gives us

$$\Psi(x, x^\alpha) = x - \int_{x^\alpha}^{x} \Psi\left(\frac{x}{y}, y\right) \frac{\mathrm{d}y}{\log y} + O\left(\frac{x}{\alpha^3 \log^2 x}\right). \tag{2.8}$$

We will continue with (2.8) to prove Theorem 6 by induction on $\lceil 1/\alpha \rceil$. The first case is $1/2 \le \alpha < 1$ and this gives

$$\begin{aligned}
\Psi(x, x^\alpha) &= x - \int_{x^\alpha}^{x} \Psi\left(\frac{x}{y}, y\right) \frac{\mathrm{d}y}{\log y} + O\left(\frac{x}{\alpha^3 \log^2 x}\right) \\
&= x - \int_{x^\alpha}^{x} \left(\frac{x}{y} - \left\{\frac{x}{y}\right\}\right) \frac{\mathrm{d}y}{\log y} + O\left(\frac{x}{\alpha^3 \log^2 x}\right) \\
&= x - x\int_{x^\alpha}^{x} \frac{\mathrm{d}y}{y\log y} + x\int_{x^\alpha}^{x} \left\{\frac{x}{y}\right\} \frac{\mathrm{d}y}{\log y} + O\left(\frac{x}{\alpha^3 \log^2 x}\right),
\end{aligned}$$

where $\{x/y\}$ denotes the fractional part of x/y. We continue with substituting $u = x/y$ and this leads to

$$\Psi(x, x^\alpha) = x - x[\log(\log y)]_{x^\alpha}^{x} + \int_1^{x^{1-\alpha}} \{u\} \frac{x}{u^2} \frac{\mathrm{d}u}{\log(x/u)} + O\left(\frac{x}{\alpha^3 \log^2 x}\right)$$

$$= x + x\log\alpha + x\int_1^{x^{1-\alpha}} \{u\} \frac{\mathrm{d}u}{u^2 \log(x/u)} + O\left(\frac{x}{\alpha^3 \log^2 x}\right)$$

$$= x\rho(1/\alpha) + \frac{x}{\log x}\int_1^{x^{1-\alpha}} \left(\frac{\{u\}}{u^2} + \frac{\{u\}\log u}{u^2\log(x/u)}\right)\mathrm{d}u + O\left(\frac{x}{\alpha^3 \log^2 x}\right).$$

We recognize that the integral comes close to

$$\int_1^\infty \frac{\{u\}}{u^2}\mathrm{d}u = \sum_{n\ge 1}\int_n^{n+1} \frac{(u-n)\mathrm{d}u}{u^2} = \sum_{n\ge 1}\left(\left(\log\frac{n+1}{n}\right) - \frac{1}{n+1}\right)$$

$$= \lim_{n\to\infty}((\log n) - (H_n - 1)) = 1 - \gamma,$$

where H_n stands for the nth harmonic number. In order to take advantage of this knowledge, we use the following bounds:

$$\frac{x}{\log x}\int_{x^{1-\alpha}}^\infty \frac{\{u\}}{u^2}\mathrm{d}u \le \frac{x}{\log x}\int_{x^{1-\alpha}}^\infty \frac{\mathrm{d}u}{u^2} = \frac{x^\alpha}{\log x}, \text{ and}$$

$$\begin{aligned}\frac{x}{\log x}\int_1^{x^{1-\alpha}}\frac{\{u\}\log u}{u^2\log(x/u)}\mathrm{d}u &\le \frac{x}{\log x}\frac{1}{\alpha\log x}\int_1^{x^{1-\alpha}}\frac{\log u}{u^2}\mathrm{d}u\\ &= \frac{x}{\alpha\log^2 x}\left[\frac{-\log u}{u}-\frac{1}{u}\right]_1^{x^{1-\alpha}}\\ &= O\left(\frac{x^\alpha}{\alpha\log x}\right)+O\left(\frac{x}{\alpha\log^2 x}\right)\\ &= O\left(\frac{x}{\alpha^3\log^2 x}\right).\end{aligned}$$

By inserting these bounds, and using $\rho(y)=1$ for $0\le y\le 1$, we have proved that

$$\Psi(x,x^\alpha)=x\rho\left(\frac{1}{\alpha}\right)+(1-\gamma)\frac{x}{\log x}\rho\left(\frac{1-\alpha}{\alpha}\right)+O\left(\frac{x}{\alpha^3\log^2 x}\right) \tag{2.9}$$

for $1/2\le\alpha<1$.

Now assume (2.9) is true for $1/m\le\alpha<1$, with m an integer >2, and let $1/(m+1)\le\alpha<1/m$. We start again with (2.8) and substitute $y=x^{1/t}$, which gives

$$\begin{aligned}\Psi(x,x^\alpha) &= x-\int_{x^\alpha}^{x}\Psi\left(\frac{x}{y},y\right)\frac{\mathrm{d}y}{\log y}+O\left(\frac{x}{\alpha^3\log^2 x}\right)\\ &= x-\int_1^{1/\alpha}\Psi(x^{(t-1)/t},x^{1/t})x^{1/t}\frac{\mathrm{d}t}{t}+O\left(\frac{x}{\alpha^3\log^2 x}\right)\\ &= x-\int_1^2\left\lfloor x^{(t-1)/t}\right\rfloor x^{1/t}\frac{\mathrm{d}t}{t}-\int_2^{1/\alpha}\Psi(x^{(t-1)/t},x^{1/t})x^{1/t}\frac{\mathrm{d}t}{t}+O\left(\frac{x}{\alpha^3\log^2 x}\right).\end{aligned} \tag{2.10}$$

Observe that we could write the Ψ-function as

$$\Psi(x^{(t-1)/t},x^{1/t})=\Psi(v,v^{1/(t-1)}),$$

with $v=x^{(t-1)/t}$. We apply the induction hypothesis (2.9) and this gives

$$\int_2^{1/\alpha}\Psi(x^{(t-1)/t},x^{1/t})x^{1/t}\frac{\mathrm{d}t}{t}=$$

$$\begin{aligned}&\int_2^{1/\alpha}\left(x\rho(t-1)+(1-\gamma)\frac{x}{\log(x^{(t-1)/t})}\rho(t-2)+O\left(\frac{x(t-1)^3}{\log^2(x^{(t-1)/t})}\right)\right)\frac{\mathrm{d}t}{t}\\ &\quad= x\int_2^{1/\alpha}\left(\rho(t-1)+\frac{(1-\gamma)\rho(t-2)}{\log(x^{(t-1)/t})}+O\left(\frac{(t-1)t^2}{\log^2 x}\right)\right)\frac{\mathrm{d}t}{t}.\end{aligned}$$

Since

$$x\int_2^{1/\alpha}O\left(\frac{(t-1)t^2}{\log^2 x}\right)\frac{\mathrm{d}t}{t}=O\left(\frac{x}{\log^2 x}\int_2^{1/\alpha}(t^2-t)\mathrm{d}t\right)=O\left(\frac{x}{\alpha^3\log^2 x}\right),$$

we get

$$\int_2^{1/\alpha} \Psi(x^{(t-1)/t}, x^{1/t}) x^{1/t} \frac{\mathrm{d}t}{t} =$$

$$x \int_2^{1/\alpha} \rho(t-1) \frac{\mathrm{d}t}{t} + (1-\gamma) \frac{x}{\log x} \int_1^{1/\alpha - 1} \rho(t-1) \frac{\mathrm{d}t}{t} + O\left(\frac{x}{\alpha^3 \log^2 x}\right). \tag{2.11}$$

By using $\rho(y) = 1$ for $0 \le y \le 1$ and substituting $u = x^{(t-1)/t}$, we obtain

$$x - \int_1^2 \left\lfloor x^{(t-1)/t} \right\rfloor x^{1/t} \frac{\mathrm{d}t}{t} = x - x \int_1^2 \rho(t-1) \frac{\mathrm{d}t}{t} + \int_1^2 \left\{ x^{(t-1)/t} \right\} x^{1/t} \frac{\mathrm{d}t}{t}$$

$$= x - x \int_1^2 \rho(t-1) \frac{\mathrm{d}t}{t} + x \int_1^{x^{1/2}} \frac{\{u\} \mathrm{d}u}{u^2 \log(x/u)}$$

$$= x - x \int_1^2 \rho(t-1) \frac{\mathrm{d}t}{t} + (1-\gamma) \frac{x}{\log x} + O\left(\frac{x}{\alpha^3 \log^2 x}\right), \tag{2.12}$$

as before. Thus, by substituting (2.11) and (2.12) into (2.10), we have

$$\Psi(x, x^\alpha) =$$

$$x - x \int_1^{1/\alpha} \rho(t-1) \frac{\mathrm{d}t}{t} + (1-\gamma) \frac{x}{\log x} - \frac{(1-\gamma)x}{\log x} \int_1^{1/\alpha - 1} \rho(t-1) \frac{\mathrm{d}t}{t} + O\left(\frac{x}{\alpha^3 \log^2 x}\right)$$

$$= x\rho\left(\frac{1}{\alpha}\right) + (1-\gamma) \frac{x}{\log x} + \frac{(1-\gamma)x}{\log x} \int_1^{1/\alpha - 1} \rho'(t) \mathrm{d}t + O\left(\frac{x}{\alpha^3 \log^2 x}\right)$$

$$= x\rho\left(\frac{1}{\alpha}\right) + (1-\gamma) \frac{x}{\log x} \rho\left(\frac{1-\alpha}{\alpha}\right) + O\left(\frac{x}{\alpha^3 \log^2 x}\right).$$

We conclude that our induction hypothesis is correct for all α and that we have proved Theorem 6. □

2.6 Proof of Theorem 7

We begin the proof with rewriting the Ψ-function of a k-semismooth number as

$$\Psi_k(x, x^{\beta_1}, \ldots, x^{\beta_k}, x^\alpha) = \sum_{x^\alpha < p_k \le x^{\beta_k}} \sum_{p_k \le p_{k-1} \le x^{\beta_{k-1}}} \cdots \sum_{p_2 \le p_1 \le x^{\beta_1}}$$

$$\# \left\{ m \le \frac{x}{p_1 \cdots p_k} : m_1 \le \left(\frac{x}{p_1 \cdots p_k}\right)^{\frac{\alpha}{1 - \frac{\log(p_1 \cdots p_k)}{\log x}}} \right\}.$$

This means that we are looking for numbers $m \in \mathbb{Z}$ such that $m \leq \frac{x}{p_1 \cdots p_k}$ and m is y-smooth with $y = x^\alpha = (x/(p_1 \cdots p_k))^{\frac{\alpha}{1-\log(p_1 \cdots p_k)/\log x}}$. Thus we can apply Theorem 6 and this gives (we omit the conditions under the summation sign)

$$\Psi_k(x, x^{\beta_1}, \ldots, x^{\beta_k}, x^\alpha) =$$

$$\sum_{p_k} \sum_{p_{k-1}} \cdots \sum_{p_1} \frac{x}{p_1 \cdots p_k} \rho\left(\frac{1 - \frac{\log(p_1 \cdots p_k)}{\log x}}{\alpha}\right) +$$

$$(1-\gamma) \sum_{p_k} \sum_{p_{k-1}} \cdots \sum_{p_1} \frac{\frac{x}{p_1 \cdots p_k}}{\log(\frac{x}{p_1 \cdots p_k})} \rho\left(\frac{1 - \frac{\log(p_1 \cdots p_k)}{\log x} - \alpha}{\alpha}\right) +$$

$$\sum_{p_k} \sum_{p_{k-1}} \cdots \sum_{p_1} O\left(\frac{\frac{x}{p_1 \cdots p_k}}{\left(\frac{\alpha}{1-(\log p_1 \cdots \log p_k)/\log x}\right)^3 \log^2\left(\frac{x}{p_1 \cdots p_k}\right)}\right). \tag{2.13}$$

We will denote the first term on the right-hand side of (2.13) with T_1. Applying Stieltjes integration gives

$$T_1 = x \int_{t_k = x^\alpha}^{x^{\beta_k}} \int_{t_{k-1}=t_k}^{x^{\beta_{k-1}}} \cdots \int_{t_1=t_2}^{x^{\beta_1}} \rho\left(\frac{1 - \frac{\log(t_1 \cdots t_k)}{\log x}}{\alpha}\right) \frac{\mathrm{d}\pi(t_1)}{t_1} \cdots \frac{\mathrm{d}\pi(t_k)}{t_k}.$$

We apply induction on the number of integrals i. In the proof we will use the Prime Number Theorem in the form $\pi(x) = \mathrm{li}(x) + \epsilon(x)$. For $i = 1$ we have

$$\int_{t_2}^{x^{\beta_1}} \rho\left(\frac{1 - \frac{\log(t_1 \cdots t_k)}{\log x}}{\alpha}\right) \frac{\mathrm{d}\pi(t_1)}{t_1} =$$

$$\int_{t_2}^{x^{\beta_1}} \rho\left(\frac{1 - \frac{\log(t_1 \cdots t_k)}{\log x}}{\alpha}\right) \frac{\mathrm{d}t_1}{t_1 \log t_1} + \int_{t_2}^{x^{\beta_1}} \rho\left(\frac{1 - \frac{\log(t_1 \cdots t_k)}{\log x}}{\alpha}\right) \frac{\mathrm{d}\epsilon(t_1)}{t_1}. \tag{2.14}$$

Applying partial integration to the last term on the right-hand side of (2.14) gives

$$\left[\rho\left(\frac{1 - \frac{\log(t_1 \cdots t_k)}{\log x}}{\alpha}\right) \frac{\epsilon(t_1)}{t_1}\right]_{t_2}^{x^{\beta_1}} - \int_{t_2}^{x^{\beta_1}} \epsilon(t_1) \frac{\mathrm{d}}{\mathrm{d}t_1}\left(\frac{1}{t_1} \rho\left(\frac{1 - \frac{\log(t_1 \cdots t_k)}{\log x}}{\alpha}\right)\right) \mathrm{d}t_1.$$

To show that these two terms are small compared to the first term on the right-hand side of (2.14), we use $\rho(x) \leq 1$ for $x \geq 0$, $\epsilon(t) = O(t/\log^c t)$ with $c = 3$ and $c = 4$, respectively, and $x^\alpha < t_2 \leq x^{\beta_2}$. So we obtain

$$\left|\left[\rho\left(\frac{1 - \frac{\log(t_1 \cdots t_k)}{\log x}}{\alpha}\right) \frac{\epsilon(t_1)}{t_1}\right]_{t_2}^{x^{\beta_1}}\right| \leq \frac{|\epsilon(x^{\beta_1})|}{x^{\beta_1}} + \frac{|\epsilon(x^\alpha)|}{x^\alpha} = O\left(\frac{1}{\alpha^3 \log^3 x}\right), \text{ and}$$

$$\int_{t_2}^{x^{\beta_1}} \epsilon(t_1)\frac{\mathrm{d}}{\mathrm{d}t_1}\left(\frac{1}{t_1}\rho\left(\frac{1-\frac{\log(t_1\cdots t_k)}{\log x}}{\alpha}\right)\right)\mathrm{d}t_1 =$$
$$O\left(\int_{t_2}^{x^{\beta_1}} \frac{t_1}{\log^4 t_1}\frac{1}{t_1^2}\rho\left(\frac{1-\frac{\log(t_1\cdots t_k)}{\log x}}{\alpha}\right)\mathrm{d}t_1\right)+$$
$$O\left(\frac{t_1}{\log^3 t_1}\frac{1}{t_1}\rho'\left(\frac{1-\frac{\log(t_1\cdots t_k)}{\log x}}{\alpha}\right)\frac{1}{\alpha t_1\log x}\mathrm{d}t_1\right).$$

We know that $x\rho'(x)+\rho(x-1)=0$ for $x\geq 1$, and we have $\frac{1-\log t/\log x}{\alpha}>0$ for $x^\alpha\leq t\leq x^\beta$. It follows that the absolute values of the ρ factor and the ρ' factor on the right-hand side are at most 1. This leads to an upper bound

$$O\left(\int_{t_2}^{x^{\beta_1}}\frac{1}{t_1\log^4 t_1}\mathrm{d}t\right)+O\left(\frac{1}{\alpha\log x}\int_{t_2}^{x^{\beta_1}}\frac{1}{t_1\log^3 t_1}\mathrm{d}t_1\right)=O\left(\frac{1}{\alpha^3\log^3 x}\right).$$

Thus we have for $i=1$

$$\int_{t_2}^{x^{\beta_1}}\rho\left(\frac{1-\frac{\log(t_1\cdots t_k)}{\log x}}{\alpha}\right)\frac{\mathrm{d}\pi(t_1)}{t_1}=$$
$$\int_{t_2}^{x^{\beta_1}}\rho\left(\frac{1-\frac{\log(t_1\cdots t_k)}{\log x}}{\alpha}\right)\frac{\mathrm{d}t_1}{t_1\log t_1}+O\left(\frac{1}{\alpha^3\log^3 x}\right).$$

Let $i_0<k-1$. Assume the following formula for all $i\leq i_0$:

$$\int_{t_{i_0+1}}^{x^{\beta_{i_0}}}\int_{t_{i_0}}^{x^{\beta_{i_0-1}}}\cdots\int_{t_2}^{x^{\beta_1}}\rho\left(\frac{1-\frac{\log(t_1\cdots t_k)}{\log x}}{\alpha}\right)\frac{\mathrm{d}\pi(t_1)}{t_1}\cdots\frac{\mathrm{d}\pi(t_{i_0})}{t_{i_0}}=$$
$$\int_{t_{i_0+1}}^{x^{\beta_{i_0}}}\int_{t_{i_0}}^{x^{\beta_{i_0-1}}}\cdots\int_{t_2}^{x^{\beta_1}}\rho\left(\frac{1-\frac{\log(t_1\cdots t_k)}{\log x}}{\alpha}\right)\frac{\mathrm{d}t_1}{t_1\log t_1}\cdots\frac{\mathrm{d}t_{i_0}}{t_{i_0}\log t_{i_0}}+$$
$$O\left(\frac{\log(\beta_{i_0-1}/\alpha)\cdots\log(\beta_1/\alpha)}{\alpha^3\log^3 x}\right). \tag{2.15}$$

Then

$$\int_{t_{i_0+2}}^{x^{\beta_{i_0+1}}}\int_{t_{i_0+1}}^{x^{\beta_{i_0}}}\cdots\int_{t_2}^{x^{\beta_1}}\rho\left(\frac{1-\frac{\log(t_1\cdots t_k)}{\log x}}{\alpha}\right)\frac{\mathrm{d}\pi(t_1)}{t_1}\cdots\frac{\mathrm{d}\pi(t_{i_0+1})}{t_{i_0+1}}=$$
$$\int_{t_{i_0+2}}^{x^{\beta_{i_0+1}}}\left(\int_{t_{i_0+1}}^{x^{\beta_{i_0}}}\cdots\int_{t_2}^{x^{\beta_1}}\rho\left(\frac{1-\frac{\log(t_1\cdots t_k)}{\log x}}{\alpha}\right)\frac{\mathrm{d}t_1}{t_1\log t_1}\cdots\frac{\mathrm{d}t_{i_0}}{t_{i_0}\log t_{i_0}}+\right.$$
$$\left.O\left(\frac{\log(\beta_{i_0-1}/\alpha)\cdots\log(\beta_1/\alpha)}{\alpha^3\log^3 x}\right)\right)\frac{\mathrm{d}\pi(t_{i_0+1})}{t_{i_0+1}}.$$

We denote the right-hand side with T_{i_0+1} and with the help of $\pi(t_{i_0+1})=\mathrm{li}(t_{i_0+1})+$

$\epsilon(t_{i_0+1})$ we get

$$T_{i_0+1} = \int_{t_{i_0+2}}^{x^{\beta_{i_0+1}}} \int_{t_{i_0+1}}^{x^{\beta_{i_0}}} \cdots \int_{t_2}^{x^{\beta_1}} \rho\left(\frac{1-\frac{\log(t_1\cdots t_k)}{\log x}}{\alpha}\right) \frac{\mathrm{d}t_1}{t_1 \log t_1} \cdots \frac{\mathrm{d}t_{i_0+1}}{t_{i_0+1}\log t_{i_0+1}}$$

$$+ \int_{t_{i_0+2}}^{x^{\beta_{i_0+1}}} \int_{t_{i_0+1}}^{x^{\beta_{i_0}}} \cdots \int_{t_2}^{x^{\beta_1}} \rho\left(\frac{1-\frac{\log(t_1\cdots t_k)}{\log x}}{\alpha}\right) \frac{\mathrm{d}t_1}{t_1 \log t_1} \cdots \frac{\mathrm{d}t_{i_0}}{t_{i_0}\log t_{i_0}} \frac{\mathrm{d}\epsilon(t_{i_0+1})}{t_{i_0+1}} +$$

$$O\left(\frac{\log(\beta_{i_0-1}/\alpha)\cdots\log(\beta_1/\alpha)}{\alpha^3 \log^3 x}\right) \int_{t_{i_0+2}}^{x^{\beta_{i_0+1}}} \frac{\mathrm{d}\pi(t_{i_0+1})}{t_{i_0+1}}. \tag{2.16}$$

For the second term on the right-hand side of (2.16) we use $\rho(x) \leq 1$ for $x \geq 0$,

$$\int_{t_{j+1}}^{x^{\beta_j}} \frac{\mathrm{d}t_j}{t_j \log t_j} = O(\log(\beta_j/\alpha)),\ (j = 1, \ldots, i_0),$$

and by applying partial integration as in the proof for $i = 1$ we obtain

$$\int_{t_{i_0+2}}^{x^{\beta_{i_0+1}}} \frac{\mathrm{d}\epsilon(t_{i_0+1})}{t_{i_0+1}} = O\left(\frac{1}{\alpha^3 \log^3 x}\right).$$

This yields that the second term on the right-hand side of (2.16) is

$$O\left(\frac{\log(\beta_{i_0}/\alpha)\cdots\log(\beta_1/\alpha)}{\alpha^3 \log^3 x}\right).$$

For the third term on the right-hand side of (2.16) we use

$$\int_{t_{i_0+2}}^{x^{\beta_{i_0+1}}} \frac{\mathrm{d}\pi(t_{i_0+1})}{t_{i_0+1}} = O\left(\int_{t_{i_0+2}}^{x^{\beta_{i_0+1}}} \frac{\mathrm{d}t_{i_0+1}}{t_{i_0+1}\log t_{i_0+1}}\right) =$$

$$O(\log(\beta_{i_0+1}/\alpha)) = O(\log(\beta_{i_0}/\alpha))$$

and this yields the same error term for the third term as we had for the second term. We conclude that (2.15) holds for all $i_0 < k$.

With (2.15) proven, we return to the expression T_1, and we apply (2.15) with $i_0 = k - 1$ to it.

$$T_1/x =$$

$$\int_{x^\alpha}^{x^{\beta_k}} \left(\int_{t_k}^{x^{\beta_{k-1}}} \cdots \int_{t_2}^{x^{\beta_1}} \rho\left(\frac{1-\frac{\log(t_1\cdots t_k)}{\log x}}{\alpha}\right) \frac{\mathrm{d}t_1}{t_1\log t_1} \cdots \frac{\mathrm{d}t_{k-1}}{t_{k-1}\log t_{k-1}} + \right.$$

$$\left. O\left(\frac{\log(\beta_{k-2}/\alpha)\cdots\log(\beta_1/\alpha)}{\alpha^3\log^3 x}\right)\right) \frac{\mathrm{d}\pi(t_k)}{t_k}.$$

We apply similar steps as in the proof of (2.15) and conclude that

$$T_1/x = \int_{x^\alpha}^{x^{\beta_k}} \int_{t_k}^{x^{\beta_{k-1}}} \cdots \int_{t_2}^{x^{\beta_1}} \rho\left(\frac{1-\frac{\log(t_1\cdots t_k)}{\log x}}{\alpha}\right) \frac{\mathrm{d}t_1}{t_1\log t_1}\cdots\frac{\mathrm{d}t_k}{t_k\log t_k} + O\left(\frac{\log(\beta_{k-1}/\alpha)\cdots\log(\beta_1/\alpha)}{\alpha^3\log^3 x}\right).$$

For the second term on the right-hand side of (2.13), denoted by T_2, we proceed in the same way. We start with Stieltjes integration, which yields

$$T_2 = (1-\gamma)x \times \int_{x^\alpha}^{x^{\beta_k}} \int_{t_k}^{\beta_{k-1}} \cdots \int_{t_2}^{x^{\beta_1}} \rho\left(\frac{1-\frac{\log(t_1\cdots t_k)}{\log x}-\alpha}{\alpha}\right) \frac{1}{\log(\frac{x}{t_1\cdots t_k})}\frac{\mathrm{d}\pi(t_1)}{t_1}\cdots\frac{\mathrm{d}\pi(t_k)}{t_k}. \quad (2.17)$$

Here we want to use a proof based on induction as well, so we fix k and denote the number of integrals with i, with $i<k$. As the idea is the same as for T_1, we will only give the induction hypothesis for all $i \le i_0 < k-1$:

$$\int_{t_{i_0+1}}^{x^{\beta_{i_0}}} \cdots \int_{t_2}^{x^{\beta_1}} \rho\left(\frac{1-\frac{\log(t_1\cdots t_k)}{\log x}-\alpha}{\alpha}\right) \frac{1}{\log(\frac{x}{t_1\cdots t_k})}\frac{\mathrm{d}\pi(t_1)}{t_1}\cdots\frac{\mathrm{d}\pi(t_{i_0})}{t_{i_0}} =$$
$$\int_{t_{i_0+1}}^{x^{\beta_{i_0}}} \cdots \int_{t_2}^{x^{\beta_1}} \rho\left(\frac{1-\frac{\log(t_1\cdots t_k)}{\log x}-\alpha}{\alpha}\right) \frac{1}{\log(\frac{x}{t_1\cdots t_k})}\frac{\mathrm{d}t_1}{t_1\log t_1}\cdots\frac{\mathrm{d}t_{i_0}}{t_{i_0}\log t_{i_0}}$$
$$+O\left(\frac{\log(\beta_{i_0-1}/\alpha)\cdots\log(\beta_1/\alpha)}{\alpha^3\log^3 x}\right). \quad (2.18)$$

Substituting (2.18) with $i_0 = k-1$ into (2.17) yields

$$T_2/((1-\gamma)x) = \int_{x^\alpha}^{x^{\beta_k}} \left(\int_{t_k}^{x^{\beta_{k-1}}} \cdots \int_{t_2}^{x^{\beta_1}} \rho\left(\frac{1-\frac{\log(t_1\cdots t_k)}{\log x}-\alpha}{\alpha}\right) \frac{1}{\log(\frac{x}{t_1\cdots t_k})} \times \frac{\mathrm{d}t_1}{t_1\log t_1}\cdots\frac{\mathrm{d}t_{k-1}}{t_{k-1}\log t_{k-1}} + O\left(\frac{\log(\beta_{i_0-1}/\alpha)\cdots\log(\beta_1/\alpha)}{\alpha^3\log^3 x}\right)\right)\frac{\mathrm{d}\pi(t_k)}{t_k}.$$

Now apply the same steps as in the proof of (2.15). This gives

$$T_2 = (1-\gamma)x \times \int_{x^\alpha}^{x^{\beta_k}} \cdots \int_{t_2}^{x^{\beta_1}} \rho\left(\frac{1-\frac{\log(t_1\cdots t_k)}{\log x}-\alpha}{\alpha}\right) \frac{1}{\log(\frac{x}{t_1\cdots t_k})}\frac{\mathrm{d}t_1}{t_1\log t_1}\cdots\frac{\mathrm{d}t_k}{t_k\log t_k}$$
$$+O\left(\frac{\log(\beta_{k-1}/\alpha)\cdots\log(\beta_1/\alpha)}{\alpha^3\log^3 x}\right).$$

This completes the second term of (2.13), and it remains to estimate the error term.

We will prove this without using induction. By Theorem 6 we have (we omit the conditions under the summation sign)

$$\sum_{p_k}\sum_{p_{k-1}}\cdots\sum_{p_1} O\left(\frac{\frac{x}{p_1\cdots p_k}}{\left(\frac{\alpha}{1-(\log p_1\cdots\log p_k)/\log x}\right)^3\log^2\left(\frac{x}{p_1\cdots p_k}\right)}\right)=$$

$$O\left(x\sum_{p_k}\frac{1}{p_k}\cdots\sum_{p_2}\frac{1}{p_2}\sum_{p_1}\frac{1}{p_1}\frac{1}{\alpha^3(1-(\beta_1+\cdots+\beta_k))^2\log^2 x}\right)=$$

$$O\left(\frac{\log(\beta_1/\alpha)\cdots\log(\beta_k/\alpha)}{\alpha^3(1-(\beta_1+\cdots+\beta_k))^2}\frac{x}{\log^2 x}\right).$$

The final step of the proof consists of substituting $t_i = x^{\lambda_i}$ for $1 \le i \le k$. □

Chapter 3

From Theory to Practice

3.1 Introduction

In this chapter we compare some of the estimates on (semi)smooth numbers from the previous chapter with data obtained by executing algorithms for factoring large numbers in order to see how useful the asymptotic estimates are for practical purposes and how large the influence of the second order term is. To do so, we need to compute the various approximations in an efficient way. The key is to have an efficient way of computing the values of the Dickman ρ function. Recall that the Dickman ρ function is the unique continuous solution of the differential-difference equation

$$\begin{cases} \rho(x) = 1 & 0 \le x \le 1 \\ \rho'(x) = -\rho(x-1)/x & x > 1. \end{cases}$$

The ρ function is piecewise analytic, and cannot be computed directly. It can be written as a Taylor series on each interval $(k-1, k]$, with k a positive integer. Patterson and Rumsey introduced the following method for calculating the values on $(k-1, k]$. Let $0 \le \xi \le 1$ and define coefficients $c_i^{(k)}$ by

$$\rho_k(k-\xi) = \sum_{i=0}^{\infty} c_i^{(k)} \xi^i.$$

Then put

$$c_0^{(1)} = 1,\ c_i^{(1)} = 0 \quad \text{for } i \ge 1$$

and for $k > 1$

$$c_0^{(k)} = \frac{1}{k-1} \sum_{j=1}^{\infty} \frac{c_j^{(k)}}{j+1},\ c_i^{(k)} = \frac{1}{i} \sum_{j=0}^{i-1} \frac{c_j^{(k-1)}}{k^{i-j}} \text{ for } i \ge 1.$$

In particular

$$c_0^{(2)} = 1 - \log 2,\ c_i^{(2)} = 1/(i2^i) \quad \text{for } i \ge 1.$$

Bach and Peralta [4] used this approach to give an efficient method for computing approximating functions for smooth and 1-semismooth numbers as given in the previous chapter. They found that 55 coefficients are enough to compute $\rho(x)$ in the range $0 \leq x \leq 20$ with a relative error of about 10^{-17}.

Lambert [25] extended the work of Bach and Peralta to 2-semismooth numbers and made a Maple [31] routine to compute the density of smooth and semismooth numbers. However, there is an error in his derivation. In §4.4 of [25] the expression for $G_2(\alpha, \beta)$ (page 97) is given as

$$G_2(\alpha,\beta) = 2\int_{2\alpha}^{\alpha+\beta}\int_{-(\mu_1-2\alpha)}^{\mu_1-2\alpha} \rho\left(\frac{1-\mu_1}{\alpha}\right)\frac{1}{(\mu_1-\mu_2)(\mu_2+\mu_1)}\mathrm{d}\mu_2\mathrm{d}\mu_1$$
$$+2\int_{\alpha+\beta}^{2\beta}\int_{-(2\beta-\mu_1)}^{2\beta-\mu_1} \rho\left(\frac{1-\mu_1}{\alpha}\right)\frac{1}{(\mu_1-\mu_2)(\mu_2+\mu_1)}\mathrm{d}\mu_2\mathrm{d}\mu_1.$$

The factor 2 in front of the two integrals should be removed. This also carries over to the subsequent derivation in §4.4 of [25].

By the theory developed in the previous chapter it is possible to calculate the second order term of the densities of 1- and 2-semismooth numbers and to calculate the density of 2-semismooth numbers with distinct upper bounds on the large primes. In Section 3.2, we show in case of Corollary 5 on 2-semismooth numbers with distinct upper bounds how to derive a numerical approximation for $x^{-1}\Psi_2(x, x^{\beta_1}, x^{\beta_2}, x^{\alpha})$ that is efficiently computable. The same ideas apply to the other theorems with a second order term.

In Section 3.3 we compare densities (computed with our program using Maple) with data from numbers factored with the multiple polynomial quadratic sieve. This will show how useful the approximations are for estimating the number of relations and how large the influence of the second order term is. Further, we give numerical evidence that using different upper bounds on the large primes can be beneficial over equal bounds. In Section 3.4 we compare our results with data obtained using the number field sieve.

3.2 Computing the approximating Ψ-function of Corollary 5

In this section we show how to derive a numerical approximation for Corollary 5 of the previous chapter. We have implemented this in Maple. Corollary 5 states for 2-semismooth numbers that if $0 < \alpha < \beta_2 < \beta_1$, $\alpha + \beta_2 + \beta_1 \leq 1$, and $x^{\alpha} \geq 2$, then for $x \to \infty$ we have

$$\Psi_2(x, x^{\beta_1}, x^{\beta_2}, x^{\alpha}) = x\int_{\alpha}^{\beta_2}\int_{\lambda_2}^{\beta_1} \rho\left(\frac{1-\lambda_1-\lambda_2}{\alpha}\right)\frac{\mathrm{d}\lambda_1}{\lambda_1}\frac{\mathrm{d}\lambda_2}{\lambda_2}$$
$$+(1-\gamma)\frac{x}{\log x}\int_{\alpha}^{\beta_2}\int_{\lambda_2}^{\beta_1} \rho\left(\frac{1-\lambda_1-\lambda_2-\alpha}{\alpha}\right)\frac{1}{1-\lambda_1-\lambda_2}\frac{\mathrm{d}\lambda_1}{\lambda_1}\frac{\mathrm{d}\lambda_2}{\lambda_2}$$

$$+O\left(\frac{\log(\beta_1/\alpha)\log(\beta_2/\alpha)}{\alpha^3(1-\beta_1-\beta_2)^2}\frac{x}{\log^2 x}\right).$$

We start with the main term in Subsection 3.2.1, and treat the second order term in Subsection 3.2.2. We (have to) disregard the error term, the size of which is expected to be, roughly spoken, $\frac{1}{\log x}$ times the size of the previous term.

3.2.1 Main term of $\Psi_2(x, x^{\beta_1}, x^{\beta_2}, x^{\alpha})$

The main term of $\Psi_2(x, x^{\beta_1}, x^{\beta_2}, x^{\alpha})$ is Mx, where

$$M := \int_{\alpha}^{\beta_2}\int_{\lambda_2}^{\beta_1} \rho\left(\frac{1-\lambda_1-\lambda_2}{\alpha}\right)\frac{\mathrm{d}\lambda_1}{\lambda_1}\frac{\mathrm{d}\lambda_2}{\lambda_2}.$$

We simplify M by rotating the area over 45° by the following substitution:

$$\begin{cases} \mu_1 = \lambda_1 + \lambda_2 \\ \mu_2 = \lambda_2 - \lambda_1. \end{cases}$$

We distinguish the cases: $2\beta_2 \le \alpha + \beta_1$ and $2\beta_2 > \alpha + \beta_1$. See Figures 3.1–3.3.

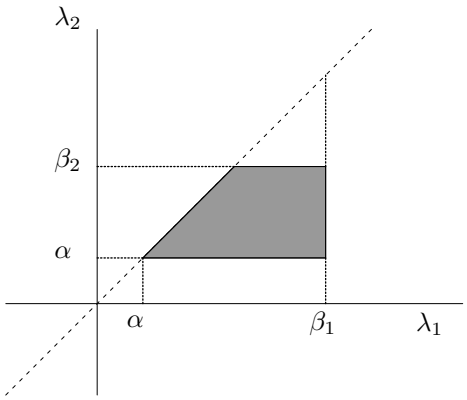

Figure 3.1: Region of integration before substitution

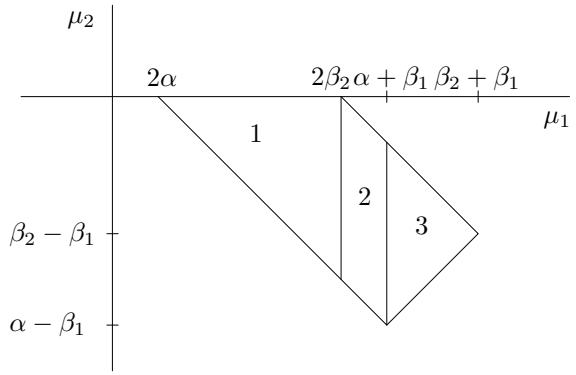

Figure 3.2: Region of integration after substitution if $2\beta_2 \leq \alpha + \beta_1$

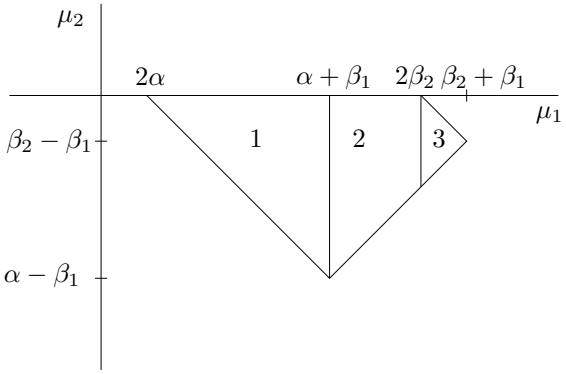

Figure 3.3: Region of integration after substitution if $2\beta_2 > \alpha + \beta_1$

In the former case, where we have $2\beta_2 \leq \alpha + \beta_1$, the substitution leads to

$$\begin{aligned} M = M_1 = 2\int_{2\alpha}^{2\beta_2}\int_{2\alpha-\mu_1}^{0} \rho\left(\frac{1-\mu_1}{\alpha}\right)\frac{1}{\mu_1^2-\mu_2^2}\mathrm{d}\mu_2\mathrm{d}\mu_1 \\ +2\int_{2\beta_2}^{\alpha+\beta_1}\int_{2\alpha-\mu_1}^{2\beta_2-\mu_1} \rho\left(\frac{1-\mu_1}{\alpha}\right)\frac{1}{\mu_1^2-\mu_2^2}\mathrm{d}\mu_2\mathrm{d}\mu_1 \\ +2\int_{\alpha+\beta_1}^{\beta_2+\beta_1}\int_{\mu_1-2\beta_1}^{2\beta_2-\mu_1} \rho\left(\frac{1-\mu_1}{\alpha}\right)\frac{1}{\mu_1^2-\mu_2^2}\mathrm{d}\mu_2\mathrm{d}\mu_1 \\ =: M_{1,1}+M_{1,2}+M_{1,3}\,, \end{aligned}$$

(cf. Figure 3.2). In the latter case ($2\beta_2 > \alpha + \beta_1$) only the boundaries of the areas, numbered 1, 2, and 3, change. We have

$$\begin{aligned} M = M_2 = 2\int_{2\alpha}^{\alpha+\beta_1}\int_{2\alpha-\mu_1}^{0} \rho\left(\frac{1-\mu_1}{\alpha}\right)\frac{1}{\mu_1^2-\mu_2^2}\mathrm{d}\mu_2\mathrm{d}\mu_1 \\ + 2\int_{\alpha+\beta_1}^{2\beta_2}\int_{\mu_1-2\beta_1}^{0} \rho\left(\frac{1-\mu_1}{\alpha}\right)\frac{1}{\mu_1^2-\mu_2^2}\mathrm{d}\mu_2\mathrm{d}\mu_1 \\ + 2\int_{2\beta_2}^{\beta_2+\beta_1}\int_{\mu_1-2\beta_1}^{2\beta_2-\mu_1} \rho\left(\frac{1-\mu_1}{\alpha}\right)\frac{1}{\mu_1^2-\mu_2^2}\mathrm{d}\mu_2\mathrm{d}\mu_1 \\ =: M_{2,1} + M_{2,2} + M_{2,3}\,, \end{aligned}$$

(cf. Figure 3.3).

We proceed in the same way as Lambert by splitting the interval of μ_1 into smaller intervals, so that we can use the approximating Taylor series of the Dickman ρ function. Let $k = \lceil (1-\mu_1)/\alpha \rceil$ and define $\xi(\mu_1) = k - (1-\mu_1)/\alpha$ as the deviation. For the complete μ_1-interval of, e.g., $M_{1,1}$ ($\mu_1 \in [2\alpha, 2\beta_2]$) we sum over all values of k with $\lceil (1-2\alpha)/\alpha \rceil \le k \le \lceil (1-2\beta_2)/\alpha \rceil$. After deriving an approximation that can be computed with, e.g., Maple, we estimate the error made when truncating the infinite sum in this derivation. For an interval $[\eta, \theta]$ of μ_1 for which k is constant, the first term ($M_{1,1}$) of M_1 becomes the sum of integrals

$$\begin{aligned} & 2\int_{\eta}^{\theta}\int_{2\alpha-\mu_1}^{0} \rho(k-\xi(\mu_1))\frac{1}{\mu_1^2-\mu_2^2}\mathrm{d}\mu_2\mathrm{d}\mu_1 \\ = \;& 2\sum_{i=0}^{\infty} c_i^{(k)} \int_{\eta}^{\theta}\int_{2\alpha-\mu_1}^{0} \xi(\mu_1)^i \frac{1}{\mu_1^2-\mu_2^2}\mathrm{d}\mu_2\mathrm{d}\mu_1 \\ = \;& 2\sum_{i=0}^{\infty} c_i^{(k)} \int_{\eta}^{\theta}\int_{2\alpha-\mu_1}^{0} \left(k - \frac{1-\mu_1}{\alpha}\right)^i \frac{1}{\mu_1^2-\mu_2^2}\mathrm{d}\mu_2\mathrm{d}\mu_1 \\ = \;& 2\sum_{i=0}^{\infty} \frac{c_i^{(k)}}{\alpha^i} \int_{\eta}^{\theta} ((k\alpha-1)+\mu_1)^i \frac{\log(\mu_1-\alpha)-\log(\alpha)}{2\mu_1}\mathrm{d}\mu_1 \\ = \;& \sum_{i=0}^{\infty} \frac{c_i^{(k)}}{\alpha^i} \int_{\eta}^{\theta} \sum_{j=0}^{i} \binom{i}{j} (k\alpha-1)^{i-j}\mu_1^{j-1}\left(\log(\mu_1-\alpha)-\log(\alpha)\right)\mathrm{d}\mu_1. \end{aligned}$$

Using $\mathrm{dilog}(x) = \int_{t=1}^{x} \frac{\log(t)}{1-t}\mathrm{d}t$, we obtain

$$\begin{aligned} \sum_{i=0}^{\infty} \frac{c_i^{(k)}}{\alpha^i} \Bigg(& (k\alpha-1)^i \left[\mathrm{dilog}\Big(\frac{\mu_1}{\alpha}\Big) + \log(\mu_1-\alpha)\log\Big(\frac{\mu_1}{\alpha}\Big) - \log(\alpha)\log(\mu_1)\right]_{\mu_1=\eta}^{\mu_1=\theta} \\ & + \sum_{j=1}^{i} \binom{i}{j} (k\alpha-1)^{i-j} \int_{\eta}^{\theta} \mu_1^{j-1}\big(\log(\mu_1-\alpha)-\log(\alpha)\big)\mathrm{d}\mu_1 \Bigg). \end{aligned}$$

In the remaining integral we substitute $\nu_1 = \mu_1 - \alpha$ and this gives

$$\int_\eta^\theta \mu_1^{j-1}\left(\log(\mu_1-\alpha)-\log(\alpha)\right)\mathrm{d}\mu_1 = \int_{\eta-\alpha}^{\theta-\alpha}(\nu_1+\alpha)^{j-1}\big(\log(\nu_1)-\log(\alpha)\big)\mathrm{d}\nu_1$$

$$= \sum_{l=0}^{j-1}\binom{j-1}{l}\alpha^{j-1-l}\int_{\eta-\alpha}^{\theta-\alpha}\nu_1^l\big(\log(\nu_1)-\log(\alpha)\big)\mathrm{d}\nu_1 \tag{3.1}$$

$$= \sum_{l=0}^{j-1}\binom{j-1}{l}\alpha^{j-(l+1)}\left[\frac{\nu_1^{l+1}\big((l+1)\log(\nu_1)-(l+1)\log(\alpha)-1\big)}{(l+1)^2}\right]_{\nu_1=\eta-\alpha}^{\nu_1=\theta-\alpha}$$

$$= \sum_{l=1}^{j}\binom{j-1}{l-1}\alpha^{j-l}\left[\frac{\nu_1^{l}\big(l\log(\nu_1)-l\log(\alpha)-1\big)}{l^2}\right]_{\nu_1=\eta-\alpha}^{\nu_1=\theta-\alpha}.$$

Combining all terms gives

$$\sum_{i=0}^{\infty}\frac{c_i^{(k)}}{\alpha^i}\Bigg((k\alpha-1)^i\left[\mathrm{dilog}\Big(\frac{x}{\alpha}\Big)+\log(x-\alpha)\log\Big(\frac{x}{\alpha}\Big)-\log(\alpha)\log(x)\right]_{x=\eta}^{x=\theta}$$

$$+\sum_{j=1}^{i}\binom{i}{j}(k\alpha-1)^{i-j}\sum_{l=1}^{j}\binom{j-1}{l-1}\alpha^{j-l}\left[\frac{x^l\big(l\log(x)-l\log(\alpha)-1\big)}{l^2}\right]_{x=\eta-\alpha}^{x=\theta-\alpha}\Bigg).$$

The second part of M_1 is treated in the same way. The only difference concerns the boundary points of the interval of μ_2. To simplify the presentation, we define

$$F(\gamma) := \sum_{i=0}^{\infty}\frac{c_i^{(k)}}{\alpha^i}\Bigg((k\alpha-1)^i\left[\mathrm{dilog}\Big(\frac{x}{\gamma}\Big)+\log(x-\gamma)\log\Big(\frac{x}{\gamma}\Big)-\log(\gamma)\log(x)\right]_{x=\eta}^{x=\theta}$$

$$+\sum_{j=1}^{i}\binom{i}{j}(k\alpha-1)^{i-j}\sum_{l=1}^{j}\binom{j-1}{l-1}\gamma^{j-l}\left[\frac{x^l\big(l\log(x)-l\log(\gamma)-1\big)}{l^2}\right]_{x=\eta-\gamma}^{x=\theta-\gamma}\Bigg).$$

We have for a part of $M_{1,2}$ with k constant

$$2\int_\eta^\theta\int_{2\alpha-\mu_1}^{2\beta_2-\mu_1}\rho(k-\xi(\mu_1))\frac{1}{\mu_1^2-\mu_2^2}\mathrm{d}\mu_2\mathrm{d}\mu_1$$

$$= \sum_{i=0}^{\infty}\frac{c_i^{(k)}}{\alpha^i}\int_\eta^\theta((k\alpha-1)+\mu_1)^i\frac{\log(\mu_1-\alpha)-\log(\alpha)-\log(\mu_1-\beta_2)+\log(\beta_2)}{\mu_1}\mathrm{d}\mu_1$$

$$= \sum_{i=0}^{\infty}\frac{c_i^{(k)}}{\alpha^i}\Bigg((k\alpha-1)^i\Bigg[\mathrm{dilog}\Big(\frac{\mu_1}{\alpha}\Big)+\log(\mu_1-\alpha)\log\Big(\frac{\mu_1}{\alpha}\Big)-\log(\alpha)\log(\mu_1)$$

$$- \left(\operatorname{dilog}\Big(\frac{\mu_1}{\beta_2}\Big) + \log(\mu_1 - \beta_2)\log\Big(\frac{\mu_1}{\beta_2}\Big) - \log(\beta_2)\log(\mu_1) \right) \Big]_{\mu_1=\eta}^{\mu_1=\theta}$$

$$+ \sum_{j=1}^{i} \binom{i}{j} (k\alpha - 1)^{i-j} \int_\eta^\theta \mu_1^{j-1} \left(\log\Big(\frac{\mu_1 - \alpha}{\alpha}) - \log(\frac{\mu_1 - \beta_2}{\beta_2} \Big) \right) \mathrm{d}\mu_1 \Big).$$

The remaining integral is split into two parts and we substitute $\nu_1 = \mu_1 - \alpha$ and $\nu_2 = \mu_1 - \beta_2$, respectively, which leads to

$$\int_\eta^\theta \mu_1^{j-1} \left(\log\Big(\frac{\mu_1 - \alpha}{\alpha}) - \log(\frac{\mu_1 - \beta_2}{\beta_2} \Big) \right) \mathrm{d}\mu_1$$

$$= \int_{\eta-\alpha}^{\theta-\alpha} (\nu_1 + \alpha)^{j-1} \log\Big(\frac{\nu_1}{\alpha}\Big) \mathrm{d}\nu_1 - \int_{\eta-\beta_2}^{\theta-\beta_2} (\nu_2 + \beta_2)^{j-1} \log\Big(\frac{\nu_2}{\beta_2}\Big) \mathrm{d}\nu_2$$

$$= \sum_{l=1}^{j} \binom{j-1}{l-1} \left(\alpha^{j-l} \left[\frac{\nu_1^l \big(l\log(\nu_1) - l\log(\alpha) - 1\big)}{l^2} \right]_{\nu_1=\eta-\alpha}^{\nu_1=\theta-\alpha} \right.$$

$$\left. - \beta_2^{j-l} \left[\frac{\nu_2^l \big(l\log(\nu_2) - l\log(\beta_2) - 1\big)}{l^2} \right]_{\nu_2=\eta-\beta_2}^{\nu_2=\theta-\beta_2} \right).$$

Combining the terms gives $F(\alpha) - F(\beta_2)$. Without further explanation we rewrite a part of $M_{1,3}$ with k constant as

$$2\int_\eta^\theta \int_{\mu_1-2\beta_1}^{2\beta_2-\mu_1} \rho\left(\frac{1-\mu_1}{\alpha}\right) \frac{1}{\mu_1^2 - \mu_2^2} \mathrm{d}\mu_2 \mathrm{d}\mu_1 = -F(\beta_2) - F(\beta_1).$$

For M_2 we only consider a part with constant k of $M_{2,2}$, as the cases $M_{2,1}$ and $M_{2,3}$ are similar to $M_{1,1}$ and $M_{1,3}$, respectively. Of course the interval for μ_1 is different, but this does not influence our treatment, as we divide the μ_1 interval into smaller pieces. Proceeding as above we find in case $M_{2,2}$ the value $-F(\beta_1)$. This concludes the treatment of the main term.

We estimate the error made by truncating the sum after the nth term. We show what happens by taking a closer look at the following expression for a typical term of $M_{1,1}$

$$2\sum_{i=0}^{\infty} c_i^{(k)} \int_\eta^\theta \int_{2\alpha-\mu_1}^{0} \xi(\mu_1)^i \frac{1}{\mu_1^2 - \mu_2^2} \mathrm{d}\mu_2 \mathrm{d}\mu_1. \tag{3.2}$$

By using $c_i^{(k)} \le (1/2)^i$ for $k \ge 2$ [4], and $\xi \le 1$, we get for the tail of (3.2)

$$\left|2\sum_{i=n+1}^{\infty} c_i^{(k)} \int_\eta^\theta \int_{2\alpha-\mu_1}^0 \xi(\mu_1)^i \frac{1}{\mu_1^2-\mu_2^2} \mathrm{d}\mu_2 \mathrm{d}\mu_1\right|$$

$$\leq 2\sum_{i=n+1}^{\infty} (1/2)^i \int_\eta^\theta \int_{2\alpha-\mu_1}^0 \frac{1}{\mu_1^2-\mu_2^2} \mathrm{d}\mu_2 \mathrm{d}\mu_1$$

$$= \frac{1}{2^{n-1}} \int_\eta^\theta \frac{\log(\mu_1-\alpha)-\log(\alpha)}{\mu_1} \mathrm{d}\mu_1$$

$$= \frac{1}{2^{n-1}} \left[\mathrm{dilog}\Big(\frac{x}{\alpha}\Big) + \log(x-\alpha)\log\Big(\frac{x}{\alpha}\Big) - \log(\alpha)\log(x)\right]_{x=\eta}^{x=\theta}. \tag{3.3}$$

As there are at most

$$\left(\left\lceil\frac{1-2\alpha}{\alpha}\right\rceil - \left\lceil\frac{1-2\beta_2}{\alpha}\right\rceil + 1\right) + \left(\left\lceil\frac{1-2\beta_2}{\alpha}\right\rceil - \left\lceil\frac{1-\alpha-\beta_1}{\alpha}\right\rceil + 1\right)$$

$$+ \left(\left\lceil\frac{1-\alpha-\beta_1}{\alpha}\right\rceil - \left\lceil\frac{1-\beta_2-\beta_1}{\alpha}\right\rceil + 1\right) = \left\lceil\frac{1}{\alpha}\right\rceil - \left\lceil\frac{1-\beta_2-\beta_1}{\alpha}\right\rceil + 1$$

$$\leq \frac{\beta_2}{\alpha} + \frac{\beta_1}{\alpha} + 2$$

subintervals, the total error is bounded by $(\beta_2/\alpha + \beta_1/\alpha + 4)$ times (3.3) for the first part. The same method applies to the other parts. In this way we find an expression for M as a finite sum of n terms with an error term for which we have a lower and upper bound.

3.2.2 Second order term of $\Psi_2(x, x^{\beta_1}, x^{\beta_2}, x^\alpha)$

We use the same idea for computing the second order term $(1-\gamma)\frac{x}{\log x}S$ as for computing the main term, where we define

$$S := \int_\alpha^{\beta_2} \int_{\lambda_2}^{\beta_1} \rho\left(\frac{1-\lambda_1-\lambda_2-\alpha}{\alpha}\right) \frac{1}{1-\lambda_1-\lambda_2} \frac{\mathrm{d}\lambda_1}{\lambda_1} \frac{\mathrm{d}\lambda_2}{\lambda_2}.$$

We rotate the integration domain and substitute $\mu_1 = \lambda_1 + \lambda_2$ and $\mu_2 = \lambda_2 - \lambda_1$. If $2\beta_2 \leq \alpha + \beta_1$, we get

$$S = S_1 = 2\int_{2\alpha}^{2\beta_2} \int_{2\alpha-\mu_1}^0 \rho\left(\frac{1-\mu_1-\alpha}{\alpha}\right) \frac{1}{(\mu_1^2-\mu_2^2)(1-\mu_1)} \mathrm{d}\mu_2 \mathrm{d}\mu_1$$

$$+ 2\int_{2\beta_2}^{\alpha+\beta_1} \int_{2\alpha-\mu_1}^{2\beta_2-\mu_1} \rho\left(\frac{1-\mu_1-\alpha}{\alpha}\right) \frac{1}{(\mu_1^2-\mu_2^2)(1-\mu_1)} \mathrm{d}\mu_2 \mathrm{d}\mu_1$$

$$+ 2\int_{\alpha+\beta_1}^{\beta_2+\beta_1} \int_{\mu_1-2\beta_1}^{2\beta_2-\mu_1} \rho\left(\frac{1-\mu_1-\alpha}{\alpha}\right) \frac{1}{(\mu_1^2-\mu_2^2)(1-\mu_1)} \mathrm{d}\mu_2 \mathrm{d}\mu_1$$

$$=: S_{1,1} + S_{1,2} + S_{1,3}.$$

If we have $2\beta_2 > \alpha + \beta_1$, only the boundaries of the three different areas change and this leads to

$$\begin{aligned} S = S_2 &= 2\int_{2\alpha}^{\alpha+\beta_1}\int_{2\alpha-\mu_1}^{0} \rho\left(\frac{1-\mu_1-\alpha}{\alpha}\right)\frac{1}{(\mu_1^2-\mu_2^2)(1-\mu_1)}\mathrm{d}\mu_2\mathrm{d}\mu_1 \\ &\quad + 2\int_{\alpha+\beta_1}^{2\beta_2}\int_{\mu_1-2\beta_1}^{0} \rho\left(\frac{1-\mu_1-\alpha}{\alpha}\right)\frac{1}{(\mu_1^2-\mu_2^2)(1-\mu_1)}\mathrm{d}\mu_2\mathrm{d}\mu_1 \\ &\quad + 2\int_{2\beta_2}^{\beta_2+\beta_1}\int_{\mu_1-2\beta_1}^{2\beta_2-\mu_1} \rho\left(\frac{1-\mu_1-\alpha}{\alpha}\right)\frac{1}{(\mu_1^2-\mu_2^2)(1-\mu_1)}\mathrm{d}\mu_2\mathrm{d}\mu_1 \\ &=: S_{2,1} + S_{2,2} + S_{2,3}. \end{aligned}$$

We show for $S_{1,1}$ how to proceed. For the other parts we only give the result. Put $l = \lceil \frac{1-\mu_1-\alpha}{\alpha} \rceil$ and define the deviation as $\xi(\mu_1) = l - \frac{1-\mu_1-\alpha}{\alpha}$. For an interval $[\eta, \theta]$ of μ_1 such that l is constant, we get

$$\begin{aligned} & 2\int_{\eta}^{\theta}\int_{2\alpha-\mu_1}^{0} \rho\big(l-\xi(\mu_1)\big)\frac{1}{(\mu_1^2-\mu_2^2)(1-\mu_1)}\mathrm{d}\mu_2\mathrm{d}\mu_1 \\ = \;& 2\sum_{i=0}^{\infty} c_i^{(l)}\int_{\eta}^{\theta}\int_{2\alpha-\mu_1}^{0} \xi(\mu_1)^i\frac{1}{(\mu_1^2-\mu_2^2)(1-\mu_1)}\mathrm{d}\mu_2\mathrm{d}\mu_1 \\ = \;& 2\sum_{i=0}^{\infty} c_i^{(l)}\int_{\eta}^{\theta}\int_{2\alpha-\mu_1}^{0} \left(l-\frac{1-\mu_1-\alpha}{\alpha}\right)^i\frac{1}{(\mu_1^2-\mu_2^2)(1-\mu_1)}\mathrm{d}\mu_2\mathrm{d}\mu_1 \\ = \;& 2\sum_{i=0}^{\infty} \frac{c_i^{(l)}}{\alpha^i}\int_{\eta}^{\theta} \big((\alpha l-1+\alpha)+\mu_1\big)^i\frac{1}{1-\mu_1}\frac{\log(\mu_1-\alpha)-\log(\alpha)}{2\mu_1}\mathrm{d}\mu_1 \end{aligned}$$

$$= \sum_{i=0}^{\infty}\frac{c_i^{(l)}}{\alpha^i}\int_{\eta}^{\theta}\sum_{j=0}^{i}\binom{i}{j}(\alpha l-1+\alpha)^{i-j}\mu_1^{j-1}\left(\frac{\log(\mu_1-\alpha)-\log(\alpha)}{1-\mu_1}\right)\mathrm{d}\mu_1.$$

Now treat $j=0$ separately to obtain

$$\begin{aligned} &\sum_{i=0}^{\infty}\frac{c_i^{(l)}}{\alpha^i}\Bigg((\alpha l-1+\alpha)^i\Big[\mathrm{dilog}\Big(\frac{\mu_1}{\alpha}\Big)+\log(\mu_1-\alpha)\log\Big(\frac{\mu_1}{\alpha}\Big)-\log(\alpha)\log(\mu_1) \\ &-\left(\mathrm{dilog}\Big(\frac{\mu_1-1}{\alpha-1}\Big)+\log(\mu_1-\alpha)\log\Big(\frac{\mu_1-1}{\alpha-1}\Big)-\log(\alpha)\log(\mu_1-1)\right)\Big]_{\mu_1=\eta}^{\mu_1=\theta} \\ &+\sum_{j=1}^{i}\binom{i}{j}(\alpha l-1+\alpha)^{i-j}\int_{\eta}^{\theta}\mu_1^{j-1}\left(\frac{\log(\mu_1-\alpha)-\log(\alpha)}{1-\mu_1}\right)\mathrm{d}\mu_1\Bigg). \end{aligned}$$

In the remaining integral we substitute $\nu_1 = \mu_1 - 1$ and this gives

$$\int_{\eta}^{\theta} \mu_1^{j-1} \left(\frac{\log(\mu_1 - \alpha) - \log(\alpha)}{1 - \mu_1} \right) \mathrm{d}\mu_1$$

$$= - \int_{\eta-1}^{\theta-1} (\nu_1 + 1)^{j-1} \left(\frac{\log(\nu_1 + 1 - \alpha) - \log(\alpha)}{\nu_1} \right) \mathrm{d}\nu_1$$

$$= - \sum_{m=0}^{j-1} \binom{j-1}{m} \int_{\eta-1}^{\theta-1} \nu_1^{m-1} \big(\log(\nu_1 + 1 - \alpha) - \log(\alpha) \big) \mathrm{d}\nu_1.$$

As before, we split the summation and this leads to

$$- \Bigg(\left[\mathrm{dilog}\Big(\frac{x}{\alpha - 1} \Big) + \log(x + 1 - \alpha) \log\Big(\frac{x}{\alpha - 1} \Big) - \log(\alpha) \log(x) \right]_{x=\eta-1}^{x=\theta-1}$$

$$+ \sum_{m=1}^{j-1} \binom{j-1}{m} \int_{\eta-1}^{\theta-1} \nu_1^{m-1} \big(\log(\nu_1 + 1 - \alpha) - \log(\alpha) \big) \mathrm{d}\nu_1 \Bigg). \tag{3.4}$$

In the integral of (3.4) we substitute $\nu_2 = \nu_1 + 1 - \alpha$ and this gives

$$\int_{\eta-\alpha}^{\theta-\alpha} (\nu_2 - 1 + \alpha)^{m-1} \big(\log(\nu_2) - \log(\alpha) \big) \mathrm{d}\nu_2$$

$$= \sum_{n=0}^{m-1} \binom{m-1}{n} (\alpha - 1)^{m-1-n} \int_{\eta-\alpha}^{\theta-\alpha} \nu_2^n \big(\log(\nu_2) - \log(\alpha) \big) \mathrm{d}\nu_2.$$

We already saw this last integral in formula (3.1). Using the expression computed there we get that (3.4) equals

$$- \Bigg(\left[\mathrm{dilog}\Big(\frac{x}{\alpha - 1} \Big) + \log(x + 1 - \alpha) \log\Big(\frac{x}{\alpha - 1} \Big) - \log(\alpha) \log(x) \right]_{x=\eta-1}^{x=\theta-1}$$

$$+ \sum_{m=1}^{j-1} \binom{j-1}{m} \sum_{n=1}^{m} \binom{m-1}{n-1} (\alpha - 1)^{m-n} \left[\frac{x^n \big(n \log(x) - n \log(\alpha) - 1 \big)}{n^2} \right]_{x=\eta-\alpha}^{x=\theta-\alpha} \Bigg).$$

Combining the terms for the first part of S_1 gives

$$\sum_{i=0}^{\infty} \frac{c_i^{(l)}}{\alpha^i} \Bigg((\alpha l - 1 + \alpha)^i \Bigg[\mathrm{dilog}\Big(\frac{x}{\alpha} \Big) + \log(x - \alpha) \log\Big(\frac{x}{\alpha} \Big) - \log(\alpha) \log(x)$$

$$- \left(\mathrm{dilog}\Big(\frac{x-1}{\alpha - 1} \Big) + \log(x - \alpha) \log\Big(\frac{x-1}{\alpha - 1} \Big) - \log(\alpha) \log(x - 1) \right) \Bigg]_{x=\eta}^{x=\theta}$$

$$- \sum_{j=1}^{i} \binom{i}{j} (\alpha l - 1 + \alpha)^{i-j} \times$$

$$\left(\left[\operatorname{dilog}\Big(\frac{x}{\alpha-1}\Big)+\log(x+1-\alpha)\log\Big(\frac{x}{\alpha-1}\Big)-\log(\alpha)\log(x)\right]_{x=\eta-1}^{x=\theta-1}+\right.$$
$$\left.\sum_{m=1}^{j-1}\binom{j-1}{m}\sum_{n=1}^{m}\binom{m-1}{n-1}(\alpha-1)^{m-n}\left[\frac{x^n\big(n\log(x)-n\log(\alpha)-1\big)}{n^2}\right]_{x=\eta-\alpha}^{x=\theta-\alpha}\right).$$

The other parts can be expressed with similar formulas, so we first define

$$G(\gamma):=\sum_{i=0}^{\infty}\frac{c_i^{(l)}}{\alpha^i}\left((\alpha l-1+\alpha)^i\left[\operatorname{dilog}\Big(\frac{x}{\gamma}\Big)+\log(x-\gamma)\log\Big(\frac{x}{\gamma}\Big)-\log(\gamma)\log(x)\right.\right.$$
$$\left.-\left(\operatorname{dilog}\Big(\frac{x-1}{\gamma-1}\Big)+\log(x-\gamma)\log\Big(\frac{x-1}{\gamma-1}\Big)-\log(\gamma)\log(x-1)\right)\right]_{x=\eta}^{x=\theta}$$
$$-\sum_{j=1}^{i}\binom{i}{j}(\alpha l-1+\alpha)^{i-j}\times$$
$$\left(\left[\operatorname{dilog}\Big(\frac{x}{\gamma-1}\Big)+\log(x+1-\gamma)\log\Big(\frac{x}{\gamma-1}\Big)-\log(\gamma)\log(x)\right]_{x=\eta-1}^{x=\theta-1}+\right.$$
$$\left.\sum_{m=1}^{j-1}\binom{j-1}{m}\sum_{n=1}^{m}\binom{m-1}{n-1}(\gamma-1)^{m-n}\left[\frac{x^n\big(n\log(x)-n\log(\gamma)-1\big)}{n^2}\right]_{x=\eta-\gamma}^{x=\theta-\gamma}\right)\right).$$

For a typical part of $S_{1,2}$ we obtain similarly $G(\alpha)-G(\beta_2)$ and a typical part of $S_{1,3}$ is $-G(\beta_2)-G(\beta_1)$.

For the second part of S_2 a typical term is $-G(\beta_1)$. The first and third parts of S_2 are similar to those of S_1.

Now we consider the tail of these expressions. As we did with the main term, we use $c_i^{(l)}\le(1/2)^i$ for $l\ge 2$ [4], and $\xi\le 1$. For $S_{1,1}$, this leads to (where n in the summation is an integer):

$$\left|2\sum_{i=n+1}^{\infty}c_i^{(l)}\int_{\eta}^{\theta}\int_{2\alpha-\mu_1}^{0}\xi(\mu_1)^i\frac{1}{(\mu_1^2-\mu_2^2)(1-\mu_1)}\mathrm{d}\mu_2\mathrm{d}\mu_1\right|$$
$$\le 2\sum_{i=n+1}^{\infty}(1/2)^i\int_{\eta}^{\theta}\int_{2\alpha-\mu_1}^{0}\frac{1}{(\mu_1^2-\mu_2^2)(1-\mu_1)}\mathrm{d}\mu_2\mathrm{d}\mu_1$$
$$=\frac{1}{2^{n-1}}\int_{\eta}^{\theta}\frac{\log(\mu_1-\alpha)-\log(\alpha)}{\mu_1(1-\mu_1)}\mathrm{d}\mu_1$$
$$=\frac{1}{2^{n-1}}\left[\operatorname{dilog}\Big(\frac{\mu_1}{\alpha}\Big)+\log(\mu_1-\alpha)\log\Big(\frac{\mu_1}{\alpha}\Big)-\log(\alpha)\log(\mu_1)\right.$$
$$\left.-\left(\operatorname{dilog}\Big(\frac{\mu_1-1}{\alpha-1}\Big)+\log(\mu_1-\alpha)\log\Big(\frac{\mu_1-1}{\alpha-1}\Big)-\log(\alpha)\log(\mu_1-1)\right)\right]_{\mu_1=\eta}^{\mu_1=\theta}.$$

The total error is bounded by $(\beta_2/\alpha+\beta_1/\alpha+2)$ times the expression above for

the first part of S_1, and the same method applies for the other parts. Therefore it suffices to take a finite sum of n terms for computing the second order term of $\Psi_2(x, x^{\beta_1}, x^{\beta_2}, x^{\alpha})$ in the required precision. In practice, we took $n = 22$, as Lambert did in his computations. (Notice that all terms in the sums in S_1 and S_2 are positive.) We verified the accuracy by varying n around $n = 22$, and found that at least 13 decimal digits are correct. In the next sections we compare results of computations based on the formulas for the main term and second order term of $\Psi_2(x, x^{\beta_1}, x^{\beta_2}, x^{\alpha})$ given above with actual data from factoring algorithms.

3.3 Experimenting with MPQS

We show in Subsection 3.3.1 how well the computation of the theoretical densities in Chapter 2 agrees with actual data sets. The data sets consist of the smooth, 1-semismooth, and 2-semismooth relations that were generated while factoring numbers with MPQS. We know how many points were sieved in total, so we can compute the (practical) densities of the different types of relations. We explicate the influence of the second order term in our computations (and neglect the O-term in the theorems). Although during the sieving many (semi)smooth numbers are collected, we do not expect that all (semi)smooth numbers are found. This is due to a practical aspect of sieving: if a polynomial value takes too much time to factor, it is put aside. This aspect is programmed as an early abort strategy. Another aspect is that a factorization attempt of a polynomial value after sieving might fail. So in advance, we expect a small discrepancy between theory and practice. Subsection 3.3.2 concerns the question how the choice between equal or different upper bounds for 2-semismooth numbers influences the number of polynomials needed for factoring the number. Subsection 3.3.3 concentrates on finding bounds which are optimal according to the theory and shows how efficient these bounds are in practice. We start with explaining how we apply the theoretical results.

As usual we have a number N to be factored, a factorbase bound F, and a large prime bound L. In case of different upper bounds for the two large primes we call the large prime bounds L_1 and L_2 with $L_1 > L_2$. We sieve the polynomial values on the interval $[-l_s, l_s]$, where l_s is chosen appropriately. We let x in $\Psi_2(x, x^{\beta_1}, x^{\beta_2}, x^{\alpha})$ be the maximum polynomial value on the sieving interval. In our implementation we have $x \approx l_s\sqrt{N/8}$, but we adjust the value of x for two reasons. The first reason concerns the smoothness of the polynomial values. The second concerns the non-linearity of the polynomial.

First, we have to take into account that we are dealing with polynomial values and they do not behave like random numbers of that size. Boender ([6], Ch. 4.4) reports a way to compare polynomial values (denoted by $W(t)$) with random values in the same interval, which we describe here briefly. It is a slightly altered version of finding the best multiplier for a given number N, as described in, e.g., [45]; the multiplier is chosen such that we get many small primes in the factorbase. The method for finding the multiplier has only small primes as input and computes the expected contribution of

these primes given N and the multiplier. The multiplier with the highest contribution gets selected. For computing the contribution of all primes in the factorbase, as these are the only primes $\leq F$ that possibly divide $W(t)$, we define n_p as the number of roots in the interval $[0, p-1]$ of the congruence relation $W(t) \equiv 0 \pmod{p}$, hence $n_p = (\frac{N}{p}) + 1$ (where $(\frac{N}{p})$ denotes the Legendre symbol and for $p = 2$ the Kronecker symbol). Using Hensel lifting, we have that any root modulo p corresponds for any $j > 1$ to a unique root mod p^j. Thus the expected factor contribution of p to $W(t)$ is

$$p^{n_p(\frac{1}{p}+\frac{1}{p^2}+\cdots)} = p^{n_p/(p-1)}.$$

If p divides N (e.g., in case of a multiplier, which is squarefree), the expected contribution is $p^{n_p/p}$. The logarithm of the total contribution of all the primes in the factorbase is therefore

$$\sum_{p \leq F, p \nmid N} n_p \frac{\log p}{p-1} + \sum_{p \leq F, p \mid N} \frac{\log p}{p}\,. \tag{3.5}$$

For random values all primes $\leq F$ are possible divisors, and the corresponding expected contribution is

$$\sum_{p \leq F} \frac{\log p}{p-1}. \tag{3.6}$$

Now we take the difference of the values, ((3.6)−(3.5)), and add this to $\log x$ (x is the maximum polynomial value on the sieving interval), as this indicates how much smoother the polynomial values are expected to be than random numbers of the same size.

The second reason concerns the definition of the Ψ-function. In the used formulas all values below x are taken into account. However, in MPQS we are dealing with points on parabolae, and this influences the value distribution. To measure this difference, we compute the average absolute value in case of a straight line and in case of a parabola. In case of a straight line with values between $-x$ and x we have an average of $x/2$. In case of a parabola we consider, without loss of generality, $f(t) = t^2 - x$ with t such that $-x \leq f(t) \leq x$. The average absolute value is computed as

$$\frac{1}{\sqrt{2x}} \left(\int_0^{\sqrt{x}} (x - t^2) \mathrm{d}t + \int_{\sqrt{x}}^{\sqrt{2x}} (t^2 - x) \mathrm{d}t \right) = \left(\frac{2\sqrt{2}}{3} - \frac{1}{3} \right) x \approx 0.6095x.$$

We conclude that on average the values on the parabola are $0.6095/0.5 \approx 1.219$ times higher than on a straight line, and therefore we multiply x by 1.219. We call this a shape factor[1].

[1]The author became aware of a different approach for computing the number of (semi)smooth numbers in the final stage of writing the thesis, and this approach may give better results. The approach is based on an article of Lenstra et al. about factoring estimates for a 1024-bit RSA modulus [29]. Instead of computing one value x for the $\Psi(x, \ldots)$ functions, the authors compute each polynomial value in the sieving area and the corresponding probability of (semi)smooth numbers. By

Now we have described how we compute x and how we correct this for the extra smoothness of the polynomial values and for the shape of the polynomial. For computing the Ψ-function we require $x^\alpha = F$ and $x^\beta = L$ (c.q. $x^{\beta_i} = L_i$ with $i = 1, 2$).

The procedure for computing the expected number of semismooth numbers is as follows. Start with computing $x = l_s\sqrt{N/8}$ and make the corrections for the primes in the factorbase and for the parabola shape. Continue with computing α and β, give these two values to our routine and evaluate the functions that give the densities of the (semi)smooth numbers. As we extended the Maple routine of Lambert, we also took his truncation bound $n = 22$ for the coefficients of rho. Experiments showed that at least 13 decimal digits are correct. For our purpose a precision of 6 digits is already enough, so it sufficed to use Lambert's bound.

Then compute $2 \times l_s \times$ #polynomials $\times$ density, which gives the expected number of (semi)smooth numbers. In the following experiments we will compare the resulting numbers with data from actual factorizations.

We start with giving the numbers that we used in our experiments, using the non-self-initializing version of Lenstra's implementation of MPQS [26] with chosen large prime bounds (overriding the packet's default large prime bounds ($\log L_1 = 1.27 \log F$, $\log L_2 = 1.2 \log F$)). We use the notation Cm to indicate a number of m digits. These numbers were generated by multiplying two primes of similar size. The numbers are:

- $C80$:= 4881523563 0968110358 9063350138 9141834760 5935453592 3573284277 7038109629 5522780973
- $C91$:= 1 4881443665 0321957176 2817556394 6548385260 9697242680 1378754699 2449766416 0616146110 8667749029
- $C100$:= 1013546883 2208255031 4768103994 6110653785 5269234453 5131324778 8905670232 8167582285 8488530234 3104491709
- $C101$:= 2 6342138102 4271344718 4607872438 8147563222 2644053886 6902044316 4108573101 5916851545 0000771213 1588359889
- $C110$:= 9252074289 8999430411 7493853880 4907942837 9538933763 7337245967 3109451808 2280847006 8920293730 1512851713 0726457659

3.3.1 Equal upper bounds

In Table 3.1 we give the different choices of the multiplier k, the number of elements of the factorbase, the factorbase bound F, the large primes bound L, and the length

integrating these probabilities over the sieving interval they get an approximation of the number of (semi)smooth numbers, which is about 10 % above the actual number of (semi)smooth numbers for the case they consider (Table 4 of [29]). In Section 3.4, where we compare the theoretical densities with the practical densities of the smooth and semismooth polynomial values in the number field sieve, we have included some experiments based on [29], namely, by a splitting of the sieving region in eight subregions and by using different approximations for the densities in each of these subregions. The results show an improvement over our initial approach (which uses one single approximation of the density for the whole sieving region).

of the sieve interval l_s, in the case of equal bounds for the large primes. Below, correction presents the value of (3.6)−(3.5), which will be added to $\log x$.

Table 3.1: Parameters used in MPQS

N	k	fbsize	F	L	l_s	correction
$C80$	1	10 970	249 797	24 979 712	374 784	−0.2676
$C91$	1	12 956	300 007	30 000 690	450 048	−0.2529
$C100$	5	15 000	351 343	35 134 320	526 848	−0.8301
$C101$	1	20 000	482 231	48 223 067	723 456	−0.0425
$C110$	11	30 000	747 853	74 785 318	1 121 792	−1.2703

We started factoring these numbers with MPQS and after 100096 polynomials we counted the number of partial relations (the corresponding polynomial values are 1-semismooth numbers) and partial-partial relations (the corresponding polynomial values are 2-semismooth numbers), and compared this with what Corollary 4 and Corollary 5 (adapted to equal upper bounds on the two large primes) of the previous chapter would imply, respectively. To see the influence of the second order term, we computed the expected number of relations according to the main term of Corollaries 4 and 5 (indicated with M in the tables below) and according to the main and second order term of these corollaries (indicated with $M + S$ in Tables 3.2 and 3.3). In parentheses we give the deviation in percents relative to the experiments.

Table 3.2: Number of partial relations after 100096 polynomials

N	experiment	Cor. 4 (M)	Cor. 4 ($M + S$)
$C80$	14 884	13 487 (−9.39 %)	14 969 (0.57 %)
$C91$	929	952 (2.48 %)	1060 (14.10 %)
$C100$	103	95 (−7.77 %)	106 (2.91 %)
$C101$	132	172 (30.30 %)	191 (44.70 %)
$C110$	43	40 (−6.98 %)	44 (2.33 %)

Table 3.3: Number of partial-partial relations after 100096 polynomials

N	experiment	Cor. 5 (M)	Cor. 5 ($M + S$)
$C80$	93 556	87 189 (−6.81 %)	96 053 (2.67 %)
$C91$	6756	7311 (8.21 %)	8086 (19.69 %)
$C100$	854	824 (−3.51 %)	914 (7.03 %)
$C101$	1391	1421 (2.16 %)	1572 (13.01 %)
$C110$	356	348 (−2.25 %)	385 (8.15 %)

Not every polynomial gives the same amount of relations, so we also executed the computation for many more polynomials. This decreases the big deviations observed when dealing with only few relations. In Table 3.4 and 3.5 we give similar information as in Table 3.2 and 3.3, but now after complete factorization.

Table 3.4: Number of partial relations

N	# pol.	experiment	Cor. 4 (M)	Cor. 4 ($M+S$)
$C80$	207 528	30 936	27 963 (−9.61 %)	31 037 (0.32 %)
$C91$	3 613 361	33 335	34 366 (3.09 %)	38 257 (14.77 %)
$C100$	33 003 742	35 914	31 171 (−13.21 %)	34 770 (−3.19 %)
$C101$	29 482 219	48 283	50 686 (4.98 %)	56 377 (16.76 %)
$C110$	161 545 944	71 504	64 719(−9.91 %)	71 905 (0.14 %)

Table 3.5: Number of partial-partial relations

N	# pol.	experiment	Cor. 5 (M)	Cor. 5 ($M+S$)
$C80$	207 528	193 898	180 769 (−6.77 %)	199 146 (2.71 %)
$C91$	3 613 361	247 315	263 906 (6.71 %)	291 899 (18.03 %)
$C100$	33 003 742	302 400	271 543 (−10.20 %)	301 164 (−0.41 %)
$C101$	29 482 219	397 119	418 551 (5.40 %)	462 917 (16.57 %)
$C110$	161 545 944	600 478	561 923 (−6.42 %)	621 129 (3.44 %)

We see that the deviation stays roughly the same after completion of the sieving. From these first experiments we conclude that the second order term for both 1-semismooth and 2-semismooth numbers adds about 10 % to the first order term.

We also notice that the experiments of $C80$, $C100$, and $C110$ agree with the theoretical values, however the deviations of the experiments of $C91$ and $C101$ are much larger.

In order to find out possible causes of the sometimes large deviations, we posed the following questions:

1. How are the polynomials chosen, especially how are the coefficients computed?
2. How does the yield of the polynomials vary as a function of the sieving time?
3. Is there some influence of the rounding off of the sieving bound?
4. Is the size of the factorbase bound important?
5. How does the choice of the multiplier and hence the primes in the factorbase influence the yield?

Below we describe our findings.

1. Choosing the polynomials

We looked into the details of the polynomial generating code of [26]. The polynomial generation is similar to the polynomial generation as explained in an article of Silverman [45]. Here, a polynomial is defined as $Q(x) = D^2x^2 + Bx + C$ with D a probable prime and $B^2 - 4D^2C = kN$, where k is the multiplier. We have no reason to believe that this way of generating polynomials leads to numbers of (semi)smooth values which on average deviate significantly from what one may expect.

2. Yield of the polynomials

As a follow-up of question 1, we looked at the yield of polynomials over time. We took the output of the factorizations of the $C91$ and the $C100$, grouped 102400 polynomials together and compared the yield of these groups in terms of full (f), partial (p), and partial-partial (pp) relations. In Table 3.6 we give some of the counts.

Table 3.6: Yield ($C91$ || $C100$)

around # pol.	f	p	pp	around # pol.	f	p	pp
1M	45	977	7077	10M	6	118	917
1M	46	940	7129	10M	8	107	947
1M	36	987	6982	10M	8	91	968
2M	46	962	7020	20M	5	89	928
2M	45	934	6948	20M	2	122	976
2M	43	981	6922	20M	2	123	892
3M	36	934	7003	30M	4	120	936
3M	47	941	6963	30M	6	134	939
3M	45	895	7045	30M	5	101	940

Note that the values of f, p and pp are of similar size. Thus to answer the second question: there is no evidence that the average yield changes significantly during the sieving.

3. Rounding off the sieving bound

The sieving is carried out in a process that crudely approximates the following idealized description. A floating point array $a[x]$, $x \in [-l_s, l_s]$, is initialized to zero. Then for each $x \in [-l_s, l_s]$, $\log(p)$ is added to those $a[x]$ for which p divides $Q(x)$, for each prime p in the factorbase. Next, those $Q(x)$-values are accepted as (semi)smooth for which the corresponding $a[x]$-values exceed the so-called sieving bound. The sieving bound is computed by dividing the maximum value of the polynomial over the sieve interval by L^2. For technical reasons the sieving bound is rounded to the nearest integer (see number between brackets). The implementation does not use the natural

logarithm. The base b used for the logarithm is $m^{1/100}$, where m is the maximal value of the polynomial over the sieve interval. This implies that $\log_b m = 100$ so that base b logarithms of all polynomial values comfortably fit in 7 bits. For the five numbers mentioned above, the sieving bounds are given in Table 3.7.

Table 3.7: Sieving bound

N	sieving bound
$C80$	65.295 (65)
$C91$	68.789 (69)
$C100$	71.371 (71)
$C101$	71.083 (71)
$C110$	73.094 (73)

The direction of the rounding does not seem to influence the deviations (notice that the expected values for both the C91 and the C101 were too high while the sieving bound for the C91 was rounded upwards and the sieving bound for the C101 downwards), but the rounding may influence the number of partial-partial relations slightly.

4. Size of the factorbase bound

We now focus on the influence of the size of the factorbase by factoring two composites of both 72 digits with different factorbase bounds.

- $C72_1 := 23\,8168522427\,1132547762\,3521738828\,0784560731\,4194384069\,3344859052\,0589460081$

- $C72_2 := 12\,3456789012\,3456789012\,3456789012\,3456789012\,3456789012\,3456798012\,3456879823$

We first give the parameters we used in these experiments and the parameters for experiments with a different multiplier (Table 3.8), and continue with the results of experiments with different factorbase sizes (Tables 3.9 and 3.10). The number in the first column identifies the experiment. In all cases we completed the factorization and give the numbers we had at the end of the factorization.

Table 3.8: Parameters used in MPQS

#	N	k	fbsize	F	L	l_s	correction
1	$C72_1$	1	5 000	102 763	10 276 294	154 112	−1.1568
2	$C72_1$	1	10 000	222 437	22 243 694	333 824	−1.1579
3	$C72_1$	1	15 000	348 421	34 842 075	522 752	−1.1552
4	$C72_2$	7	5 000	105 361	10 536 104	158 208	−0.9975
5	$C72_2$	7	10 000	223 441	22 344 100	335 360	−1.0081
6	$C72_2$	7	15 000	348 433	34 843 284	522 752	−1.0105
7	$C72_1$	5	5 000	103 093	10 309 305	154 624	1.3074
8	$C72_1$	13	5 000	103 573	10 357 307	155 136	0.6836

Table 3.9: Number of partial relations at the end

#	# pol.	experiment	Cor. 4 (M)	Cor. 4 ($M+S$)
1	58 403	14 109	10 874 (−22.93 %)	12 134 (−14.00 %)
2	17 401	30 801	24 967 (−18.94 %)	27 607 (−10.37 %)
3	9 189	48 301	39 559 (−18.10 %)	43 534 (−9.87 %)
4	93 740	13 617	15 025 (10.34 %)	16 768 (23.14 %)
5	26 451	29 734	31 461 (5.81 %)	34 800 (17.04 %)
6	13 614	47 367	48 568 (2.54 %)	53 469 (12.88 %)

Table 3.10: Number of partial-partial relations at the end

#	# pol.	experiment	Cor. 5 (M)	Cor. 5 ($M+S$)
1	58 403	85 454	71 559 (−16.26 %)	79 167 (−7.36 %)
2	17 401	156 339	133 558 (−14.57 %)	146 406 (−6.35 %)
3	9 189	218 140	189 042 (−13.34 %)	206 241 (−5.45 %)
4	93 740	90 495	99 976 (10.48 %)	110 630 (22.25 %)
5	26 451	164 351	171 121 (4.12 %)	187 670 (14.19 %)
6	13 614	230 417	236 223 (2.52 %)	257 841 (11.90 %)

We see that the deviation becomes less when we increase the factorbase size and this can be explained by the fact that we use an asymptotic function for $x \to \infty$. If the factorbase is made larger, then the sievelength increases and this increases the polynomial values, which are substituted in the Ψ-function. Another consequence of increasing the sievelength is that we find more relations per polynomial so that we need fewer polynomials, even though we need more relations to cover all primes in

the larger factorbase. Moreover, we can adjust the choice of the polynomials to the sievelength so that the (absolute) polynomial values become smaller on average and generate more (semi)smooth numbers.

5. Multiplier / Primes in the factorbase

To observe the influence of the multiplier and subsequently the primes in the factorbase, we repeated experiment 1, but with multipliers 5 and 13. The results are in Tables 3.11 and 3.12.

Table 3.11: Number of partial relations at the end

#	# pol.	experiment	Cor. 4 (M)	Cor. 4 ($M+S$)
1	58 403	14 109	10 874 (−22.93 %)	12 134 (−14.00 %)
7	182 537	13 008	13 898 (6.84 %)	15 535 (19.43 %)
8	142 104	13 366	11 410 (−14.63 %)	12 752 (−4.59 %)

Table 3.12: Number of partial-partial relations at the end

#	# pol.	experiment	Cor. 5 (M)	Cor. 5 ($M+S$)
1	58 403	85 454	71 559 (−16.26 %)	79 167 (−7.36 %)
7	182 537	92 114	98 221 (6.63 %)	108 892 (18.21 %)
8	142 104	90 778	80 275 (−11.57 %)	88 982 (−1.98 %)

We know that the multiplier influences which primes are in the factorbase and this affects the correction we apply (cf. Table 3.8, last column). However, we do not know how to predict the influence of the primes in the factorbase.

After looking at all these different aspects of sieving, we can only give a partial explanation for the deviation between theoretical and practical results. It is closely related with the set of primes that are in the factorbase, and although we apply a correction for this, our approach of using one value to represent all polynomial values seems to be too crude.

3.3.2 Different upper bounds

Compared with the first five experiments of Subsection 3.3.1, we only change the large prime bound L to a bound L_1 for the largest prime and a bound L_2 for the second largest prime in the case of 2-semismooth numbers. For 1-semismooth numbers we take the bound L_1. The correction factor does not change, as we keep the same

factorbase bound. The large prime bounds used in the following experiments are given in Table 3.13.

Table 3.13: Parameters used in MPQS

N	F	L_2	L_1	l_s
$C80$	249 797	12 489 854	49 959 435	374 784
$C91$	300 007	15 000 354	60 001 434	450 048
$C100$	351 343	17 567 141	70 268 609	526 848
$C101$	482 231	24 111 564	96 446 162	723 456
$C110$	747 853	37 392 644	149 570 695	1 121 792

We compare the theoretical results with the sieving data of the experiments after 100096 polynomials (Tables 3.14 and 3.15) and after many more polynomials (Tables 3.16 and 3.17). A $*$ indicates that the factorization was complete; we expect small changes in the percentages deviation when the number is not completely factored.

Table 3.14: Number of partial relations after 100096 polynomials

N	experiment	Cor. 4 (M)	Cor. 4 ($M+S$)
$C80$	18 578	16 847 ($-9.32,\%$)	18 694 (0.62 %)
$C91$	1148	1190 (3.66 %)	1324 (15.33 %)
$C100$	144	118 (-18.06 %)	132 (-8.33 %)
$C101$	173	215 (24.28 %)	239 (38.15 %)
$C110$	62	50 (-19.35 %)	56 (-9.68 %)

Table 3.15: Number of partial-partial relations after 100096 polynomials

N	experiment	Cor. 5 (M)	Cor. 5 ($M+S$)
$C80$	123 775	116 261 (-6.07 %)	128 047 (3.45 %)
$C91$	8973	9762 (8.79 %)	10 795 (20.31 %)
$C100$	1177	1102 (-6.37 %)	1222 (3.82 %)
$C101$	1863	1899 (1.93 %)	2100 (12.72 %)
$C110$	489	465 (-4.91 %)	514 (5.11 %)

Table 3.16: Number of partial relations

N	# pol.	experiment	Cor. 4 (M)	Cor. 4 ($M+S$)
$C80*$	202 630	37 677	34 105 (−9.48 %)	37 845 (0.45 %)
$C91$	526 592	6031	6258 (3.76 %)	6965 (15.49 %)
$C100$	512 000	697	606 (−13.06 %)	676(−3.01 %)
$C101$	9 606 400	19 530	20 634 (5.65 %)	22 947 (17.50 %)
$C110$	36 312 576	20 252	18164 (−10.31 %)	20180 (−0.36 %)

Table 3.17: Number of partial-partial relations

N	# pol.	experiment	Cor. 5 (M)	Cor. 5 ($M+S$)
$C80*$	202 630	250 264	235 353 (−5.96 %)	259 213 (3.58 %)
$C91$	526 592	47 519	51 356 (8.07 %)	56 792 (19.51 %)
$C100$	512 000	6158	5639 (−8.43 %)	6254(1.56 %)
$C101$	9 606 400	172 217	182 237 (5.82 %)	201 520 (17.02 %)
$C110$	36 312 576	179 687	168784 (−6.07 %)	186548 (3.82 %)

Observe that we need fewer polynomials to factor the $C80$ when we choose different upper bounds, and we expect that this is also true for other numbers. Nevertheless, we see the same behavior in deviation as in the experiments of Subsection 3.3.1. This is related to the factorbase and the sievelength, which remained the same in these two sets of experiments. In the next subsection we will describe more experiments with different upper bounds.

3.3.3 Optimal bounds

We would like to compute optimal bounds for the factorbase and the large primes, in the sense of getting higher densities $(\Psi(x,\ldots)/x)$ for the three different types of relations. Note that a higher density does not guarantee a shorter sieving time. Increasing the factorbase bound (F) and the large prime bounds (L_1, L_2) will increase the densities, but we will need more relations to find a dependency. Additionally, the sieving time per relation will change. A short sieving test can provide insight into the sieving times.

To start with an easy case, we keep F and the product L_1L_2 constant and only vary L_1 and L_2. If L_2 becomes much smaller, then factoring the cofactor (the product of two primes, which is left after sieving) will take less time. For now, we only concentrate on the densities we get. In order to be able to compare theoretical and practical results, we take the second experiment of Table 3.8 (a 72-digit number) as our

starting point. We first give several combinations of L_1 and L_2 and the corresponding densities for partial and partial-partial relations.

Table 3.18: Densities

L_2	L_1	density p	density pp
22 243 694	22 243 694	2.37631×10^{-6}	1.26019×10^{-5}
13 979 427	35 393 616	2.74766×10^{-6}	1.55948×10^{-5}
8 785 603	56 317 416	3.15301×10^{-6}	1.74530×10^{-5}
5 521 458	89 610 830	3.59565×10^{-6}	1.82767×10^{-5}
3 470 052	142 586 459	4.07918×10^{-6}	1.81449×10^{-5}
2 180 812	226 879 923	4.60754×10^{-6}	1.71180×10^{-5}
1 370 568	361 005 526	5.18506×10^{-6}	1.52387×10^{-5}

As expected, the density of the partial relations increases as L_1 increases; the density of the partial-partial relations first increases and then decreases again (as L_2 is getting closer to F). In Table 3.19, we compare these theoretical densities with the number of relations in an actual data set and keep track of the time it takes to find all relations (including full relations).

Table 3.19: Number of relations and time

L_2	L_1	# pol	# p	# pp	time(sec.)
22 243 694	22 243 694	17 401	30 801	156 339	965.05
16 682 765	29 657 528	17 213	33 317	178 161	941.65
11 121 843	44 487 403	17 017	37 227	200 652	936.20
5 520 883	89 610 997	16 963	45 196	218 746	938.73
2 780 463	177 949 470	17 188	55 302	215 497	943.62

The experiment with $L_2 \approx 5.52$M has the highest theoretical density and this agrees with the actual data set as this set has the highest number of pp's and the least number of polynomials. Although these parameters gave the highest density, the (actual) experiment with $L_2 \approx 11.12$M gave a slightly lower time. This is due to performing fewer factorizations on the pp's. Another aspect concerning time is the implementation aspect of how the cofactor is factored, which becomes important if L_2 is much smaller than L_1 and different methods for factoring the cofactor are used.

We computed for more combinations N, F, L_1, and L_2 the densities and compared it with the sieving time. This showed the same pattern as described above.

In order to make a good prediction of the time, we first should be able to determine how many relations we need to factor the number. Combined with the theoretical densities we can determine the number of polynomials and this will give an estimate

of the total time, as a short sieving test will tell how long a few polynomials take.

For the moment, choosing L_1 and L_2 when the density pp in Table 3.18 is maximal seems a good choice once the factorbase and L_1L_2 are chosen. For choosing an optimal factorbase bound we refer to [28] and Chapter 6 of [14]. For choosing L_1L_2 we can compute some densities after the factorbase bound is fixed and use the product with the optimal density.

3.4 Experimenting with NFS

We now compare the theoretical densities with the practical densities of the semismooth numbers in the case of the number field sieve. This is more complicated than the comparison for MPQS, as there are more aspects that we must take into account. We start with a similar approach as in Subsection 3.3.1, but we give as well the better approach (already mentioned in a footnote in Section 3.3) that takes more polynomial values into account (as in [29]) by dividing the sieving region into subregions.

In NFS we work with two homogeneous polynomials $F_1(x, y)$ and $F_2(x, y)$, and we search for pairs (a, b) with $\gcd(a, b) = 1$ and $b > 0$ such that both $|F_1(a, b)|$ and $|F_2(a, b)|$ are (semi)smooth. For both polynomials we are looking for the highest polynomial value in the sieving region and if necessary we apply a correction for the average value and the roots of $F_i(a, b)$, for $i = 1, 2$, modulo primes in the factorbase (similar to the correction we applied for the polynomials in MPQS). We assume that the probability of $F_1(a, b)$ being semismooth is independent of the probability of $F_2(a, b)$ being semismooth.

First we give our results for a number $(19,183-, C131)$ we factored with the special number field sieve. The polynomials are

- $F_1(x, y) = 37589973457545958193355601x - y$, and
- $F_2(x, y) = x^6 + 19x^3y^3 + 361y^6$,

and the sieving bounds are $F = 30\text{M}$ and $L = 250\text{M}$. The sieving region is described by $x \in [-1.75\text{M}, 1.75\text{M}]$ and $y \in [1, 700\,000]$. To compute the correction for the shape of the polynomial, we start with computing the average value of $F_2(x, y)$:

$$\frac{1}{3.5\text{M} \times 700\,000} \int_1^{700\,000} \int_{-1.75\text{M}}^{1.75\text{M}} \left|x^6 + 19x^3y^3 + 361y^6\right| \mathrm{d}x\, \mathrm{d}y \approx 1.017 \times 10^{37}.$$

The maximum value occurs at the border of the sieving region with $F_2(1.75\text{M}, 700\,00) \approx 1.0612 \times 10^{38}$; the average value is 0.096 times the maximum value. Compared with a straight line, we expect that on average the values of $F_2(x, y)$ are $\frac{0.5}{0.096} \approx 5.217$ times smaller; we call this a shape factor.

To take the root properties into account, we compute the following (see [35],Ch. 3): for all primes p in the factorbase that do not divide the discriminant Δ of $F_2(x, 1)$ we add $\left(1 - \frac{n_p p}{p+1}\right) \frac{\log p}{p-1}$ as correction, where n_p is the number of distinct roots of

$F_2(x,1) \mod p$. For primes that divide Δ we computed with a numerical simulation the average contribution of these primes for a sample of 10^6 values of $F_2(x,y)$. The discriminant of $F_2(x,1)$ factors as $-3^9\,19^{10}$. When we evaluate this polynomial for $0 \le x \le 999$ and $1 \le y \le 999$ we get on average 0.250 times a factor 3 in the polynomial value. As correction for the prime 3 we take $(1/2 - 0.250)\log(3)$. For the prime 19 we computed a correction of $(1/18 - 0.099)\log(19)$. The total correction for primes in the factorbase, including primes that divide the discriminant is 0.601.

We compute the value x which will be substituted in the $\Psi(x,...)$-function as follows: $\log(x) = \log(1.0612 \times 10^{38}) - \log(5.217) + 0.601 \approx 86.508$. This leads to $\alpha = \log(30\text{M})/\log(x) = 0.199$ and $\beta = \log(250\text{M})/\log(x) = 0.224$ for $F_2(x,y)$. For the linear polynomial $F_1(x,y)$ we compute the maximum value, which turns out to be $\approx 6.578 \times 10^{31}$. We do not need a correction for the shape of the polynomial nor a correction for root properties, thus we get $\alpha = 0.235$ and $\beta = 0.264$ for $F_1(x,y)$.

In order to compute the number of (semi)smooth numbers, we divide the relations into different types, based on the number of large primes on each side. For $i,j \in \{0,1,2\}$ we denote with $r_i a_j$ the set of relations with i large primes on the rational side and j large primes on the algebraic side. Now we approximate $\#r_i a_j$ by

$$\frac{6}{\pi^2} \times 3.5\text{M} \times 700\,000 \times \frac{\Psi_i(x,...)}{x} \times \frac{\Psi_j(x,...)}{x},$$

where we take the appropriate Ψ-function for each relation set, and include the second order term of each Ψ-function in our computation. Note that the fraction $6/\pi^2$ comes from the requirement $\gcd(a,b) = 1$ as this fraction expresses the probability that this is the case for two random numbers a and b.

Our theoretically expected numbers and the numbers of (semi)smooth numbers of the actual data set are in Table 3.20. For the algebraic side we took into account that the actual sieving used $F^{0.1}L^{1.9}$ as upper bound for the product of the two large primes in 2-semismooth numbers, so we use the square root of this number as bound on each of the two large primes. This led to $\beta = 0.222$.

Table 3.20: Number of relations $r_i a_j$

relation type	theory	actual
$r_0 a_0$	1 447 297	2 040 867
$r_0 a_1$	2 712 542	3 524 813
$r_0 a_2$	1 550 396	2 027 945
$r_1 a_0$	2 072 000	2 925 595
$r_1 a_1$	3 882 206	5 067 356
$r_1 a_2$	2 218 936	2 922 010
$r_2 a_0$	912 503	1 280 620
$r_2 a_1$	1 709 713	2 224 917
$r_2 a_2$	977 213	1 286 313

We see relatively big differences. However, the ratios of actual and theoretical numbers as ordered in Table 3.21 show some regularities.

Table 3.21: Ratio of actual and computed numbers per set

	r_0	r_1	r_2
a_0	1.410	1.412	1.403
a_1	1.299	1.305	1.301
a_2	1.308	1.317	1.316

The horizontal deviation is rather small. It seems that there is a systematic deviation in our estimation of the value x on the algebraic side. With reverse computing we found that if we had $\log(x) = 84.344$, our theoretical number of relations would be close to the actual number of relations.

However, as already mentioned in Subsection 3.3.1, the author became aware of a different approach for computing the number of (semi)smooth numbers in the final stage of writing the thesis. The approach is described in an article of Lenstra et al. about factoring estimates for a 1024-bit RSA modulus [29]. We think that this approach will give better results, as all polynomial values are computed, instead of only one value. To test this assumption, we start with dividing the integration region in eight equal regions and apply to each region the same steps as we did for the complete region. The regions are given in the following table.

Table 3.22: Boundaries of the eight regions

region	x	y
1	[-1 750 000 , -875 000]	[350 000, 700 000]
2	[-875 000 , 0]	[350 000, 700 000]
3	[0 , 875 000]	[350 000, 700 000]
4	[875 000 , 1 750 000]	[350 000, 700 000]
5	[-1 750 000 , -875 000]	[1 , 350 000]
6	[-875 000 , 0]	[1 , 350 000]
7	[0 , 875 000]	[1 , 350 000]
8	[875 000 , 1 750 000]	[1 , 350 000]

In the next table we give for each region the minimum and maximum value of $F_2(x, y)$, the average value of $F_2(x, y)$, and the factor for the shape of the polynomial compared with a straight line.

Table 3.23: $F_2(x, y)$ characteristics per region

region	$\min(F_2(x, y))$	$\max(F_2(x, y))$	average	shape factor
1	5.667×10^{35}	3.855×10^{37}	3.672×10^{37}	0.517
2	4.977×10^{35}	4.247×10^{37}	1.159×10^{37}	1.810
3	6.636×10^{35}	4.729×10^{37}	1.262×10^{37}	1.848
4	1.658×10^{36}	1.061×10^{38}	2.786×10^{37}	1.875
5	3.366×10^{35}	2.872×10^{37}	7.726×10^{36}	1.837
6	271	6.636×10^{35}	1.248×10^{35}	2.659
7	361	1.658×10^{36}	1.930×10^{35}	4.295
8	4.488×10^{35}	3.375×10^{37}	8.749×10^{36}	1.903

For the linear polynomial we give the minimum and maximum for each region as well.

Table 3.24: $F_1(x, y)$ characteristics per region

region	$\min(F_1(x, y))$	$\max(F_1(x, y))$
1	3.289×10^{31}	6.578×10^{31}
2	350 000	3.289×10^{31}
3	350 000	3.289×10^{31}
4	3.289×10^{31}	6.578×10^{31}
5	3.289×10^{31}	6.578×10^{31}
6	1	3.289×10^{31}
7	1	3.289×10^{31}
8	3.289×10^{31}	6.578×10^{31}

When we use these values to compute the corresponding densities for each region and add the theoretical number of (semi)smooth relations, we find for the total region the following number of relations:

Table 3.25: Number of relations $r_i a_j$

relation type	theory	actual	ratio
r_0a_0	1 843 020	2 040 867	1.107
r_0a_1	3 355 199	3 524 813	1.050
r_0a_2	1 848 677	2 027 945	1.097
r_1a_0	2 472 207	2 925 595	1.183
r_1a_1	4 501 569	5 067 356	1.126
r_1a_2	2 481 009	2 922 010	1.178
r_2a_0	1 020 612	1 280 620	1.255
r_2a_1	1 858 914	2 224 917	1.197
r_2a_2	1 024 899	1 286 313	1.255

This is already much better than our first approach (Table 3.20) with only value for approximating the entire region. It remains to be seen how far we should go with subdividing our regions into smaller regions to get an even better result. To see if this behavior is consistent, we performed the computations for another factorization. However, even after further subdividing we probably still have some difference, as in practice less effort is spend on relations of type r_2a_1 and r_2a_2.

This time we factored $26,142+$, $C124$ with the general number field sieve. The polynomials and sieving parameters are:

- $F_1(x,y) = x - 421022476971059394365 88y$
- $F_2(x,y) = 8848132902x^5 - 10040121975867x^4y -$
  $18557337266133130x^3y^2 + 438450176574957877 42x^2y^3 +$
  $384219666742699491428188xy^4 + 321320915029372552455813365y^5$
- $F = 30\text{M}$
- $L = 250\text{M}$
- Sieving region: $x \in [-200\text{M}, 200\text{M}]$ and $y \in [1, 88000]$.

The maximum value of $F_2(x,y)$ is $F_2(200\text{M}, 88\,000) \approx 7.767 \times 10^{51}$, and the average value is

$$\frac{1}{400\text{M} \times 88\,000} \int_1^{700\,000} \int_{-200\text{M}}^{200\text{M}} |F_2(x,y)| \,\mathrm{d}x \,\mathrm{d}y \approx 8.852 \times 10^{50}.$$

We apply a correction of $0.5/0.114 \approx 4.387$ for the shape of this polynomial.

For the root properties we first look at the discriminant Δ, which factors as $\Delta = 2^9\, 3^4\, 13^2\, 31\, 73\, C140$. As we sieve with primes up to 30M, we used trial division up to 30M to ensure that the factors given are the only factors below 30M. For the primes 2, 3, 13, 31, and 73 we computed the average contribution of these primes. For the

other primes p in the factorbase we used again $\left(1-\frac{n_p p}{p+1}\right)\frac{\log p}{p-1}$ as correction. For the prime divisors of the leading coefficient of $F_2(x,y)$, 7, 17, and 44417, we increase n_p with 1, as these primes divide the leading coefficient of F_2, which leads to a projective root. The total correction is

$$-3.019 + (1-2.332)\log(2) + \left(\frac{1}{2} - 1.875\right)\log(3) + \left(\frac{1}{12} - 0.232\right)\log(13)$$
$$+ \left(\frac{1}{30} - 0.128\right)\log(31) + \left(\frac{1}{72} - 0.014\right)\log(73) = -6.160.$$

Combining all corrections gives

$$\log(x) = \log(7.767 \times 10^{51}) - \log(4.387) - 6.160 = 111.843,$$

and $\alpha = 0.154$, and $\beta = 0.173$. For the linear polynomial we have $\log(x) = 63.479$, $\alpha = 0.271$, and $\beta = 0.305$. We compute

$$r_i a_j = \frac{6}{\pi^2} \times 400\text{M} \times 88\,000 \times \frac{\Psi_i(x,...)}{x} \times \frac{\Psi_j(x,...)}{x},$$

and the results are given in Table 3.26. As with $19,183-$, we take the different upper bound for 2-semismooth numbers into account, which gives $\beta = 0.172$.

Table 3.26: Number of relations $r_i a_j$

relation type	theory	actual
$r_0 a_0$	1 099 680	1 857 736
$r_0 a_1$	3 073 751	4 729 263
$r_0 a_2$	2 910 129	3 535 594
$r_1 a_0$	1 241 976	2 144 878
$r_1 a_1$	3 471 488	5 465 241
$r_1 a_2$	3 286 693	4 059 856
$r_2 a_0$	379 158	671 005
$r_2 a_1$	1 059 798	1 713 980
$r_2 a_2$	1 003 382	1 260 477

If we look at the ratio (Table 3.27), we get

Table 3.27: Ratio of actual and computed numbers per set

	r_0	r_1	r_2
a_0	1.689	1.727	1.770
a_1	1.539	1.574	1.617
a_2	1.215	1.235	1.256

Again the horizontal variation is rather small. Our attempt to find a better value x with reverse computing, as we did with the previous example, indicates that $\log(x)$ should be in the neighbourhood of 109.036.

Here we decided as well to divide the entire region in eight equal regions and compute per region the theoretical number of relations. The regions are given in the following table.

Table 3.28: Boundaries of the eight regions

region	x	y
1	[-200 000 000 , -100 000 000]	[44 000, 88 000]
2	[-100 000 000 , 0]	[44 000, 88 000]
3	[0 , 100 000 000]	[44 000, 88 000]
4	[100 000 000 , 200 000 000]	[44 000, 88 000]
5	[-200 000 000 , -100 000 000]	[1 , 44 000]
6	[-100 000 000 , 0]	[1 , 44 000]
7	[0 , 100 000 000]	[1 , 44 000]
8	[100 000 000 , 200 000 000]	[1 , 44 000]

In the next table we give for each region the minimum and maximum value of $F_2(x, y)$, the average value of $F_2(x, y)$, and the factor for the shape of the polynomial compared with a straight line.

Table 3.29: $F_2(x, y)$ characteristics per region

region	$\min(F_2(x, y))$	$\max(F_2(x, y))$	average	shape factor
1	1.504×10^{50}	4.813×10^{51}	1.482×10^{51}	1.573
2	0	1.696×10^{51}	2.336×10^{50}	3.629
3	5.299×10^{49}	4.155×10^{51}	1.030×10^{51}	1.991
4	2.427×10^{50}	7.767×10^{51}	2.426×10^{51}	1.551
5	8.848×10^{49}	3.336×10^{51}	1.034×10^{51}	1.571
6	0	1.504×10^{50}	2.164×10^{49}	3.474
7	3.213×10^{26}	2.427×10^{50}	3.368×10^{49}	3.603
8	7.151×10^{49}	2.831×10^{51}	8.210×10^{50}	1.681

For the linear polynomial F_1 we give the minimum and maximum for each region as well.

Table 3.30: $F_1(x, y)$ characteristics per region

region	$\min(F_1(x, y))$	$\max(F_1(x, y))$
1	1.853×10^{27}	3.705×10^{27}
2	1.853×10^{27}	3.705×10^{27}
3	1.853×10^{27}	3.705×10^{27}
4	1.853×10^{27}	3.705×10^{27}
5	4.210×10^{22}	1.853×10^{27}
6	4.210×10^{22}	1.853×10^{27}
7	4.210×10^{22}	1.853×10^{27}
8	4.210×10^{22}	1.853×10^{27}

When we use these values to compute the corresponding densities for each region and add the theoretical number of (semi)smooth relations, we find for the total region the following number of relations:

Table 3.31: Number of relations $r_i a_j$

relation type	theory	actual	ratio
$r_0 a_0$	1 410 985	1 857 736	1.317
$r_0 a_1$	3 868 632	4 729 263	1.122
$r_0 a_2$	3 579 839	3 535 594	0.988
$r_1 a_0$	1 574 800	2 144 878	1.362
$r_1 a_1$	4 318 392	5 465 241	1.266
$r_1 a_2$	3 996 721	4 059 856	1.016
$r_2 a_0$	472 240	671 005	1.421
$r_2 a_1$	1 295 250	1 713 980	1.323
$r_2 a_2$	1 199 089	1 260 477	1.051

Again, this is already much better than our first approach with only value for approximating the entire region. As the region around $(x, y) = (0, 1)$ (regions 6 and 7) has smaller polynomial values, the chance of finding relations with at most one large prime on the algebraic side will increase. It is likely that dividing these two regions in smaller pieces will give better approximations, but further research is needed to confirm this.

Chapter 4

Predicting the Sieving Effort for the Number Field Sieve

4.1 Introduction

A popular method for factoring large numbers is the number field sieve [27], as this is the fastest algorithm known for numbers of at least 90 digits. In order to estimate the most time-consuming step of this method, namely the sieving step in which the relations are generated, one compares with actual sieving times for numbers of comparable size. If these are not available, one may try to extrapolate actual sieving times for smaller numbers, using the formula for the running time $L(N)$ of this method, where N is the number to be factored. We have

$$L(N) = \exp(((64/9)^{1/3} + o(1))(\log N)^{1/3}(\log\log N)^{2/3}), \text{ as } N \to \infty,$$

where the logarithms are natural [27]. These estimates (for computations we disregard the $o(1)$ and get a heuristic value) can be 10–30 % off from the real running times.

After the sieving we continue with the linear algebra step. During this step we have to find dependencies in a matrix, where each row corresponds with a relation and each column with a rational prime $\leq L$ or an algebraic prime with norm $\leq L$ such that all relations are represented by a row and all primes which occur in the relations are represented as a column. If a prime occurs an odd number of times in a relation, we put a one as entry of the corresponding row and column, and a zero otherwise. After representing all relations in the matrix, we remove those relations with a 1 that is the only 1 in the entire column, the so-called singletons. This may generate new singletons, so this singleton removal step is repeated until all columns contain at least two 1's. In practice, the singleton removal is done before actually building the matrix.

During the sieving step, the number of useful relations (those which remain after

singleton removal) grows in a hard-to-predict way as a function of the number of relations found, as the primes of newly found relations may or may not match with the primes of earlier found relations. This growth behavior differs from number to number, which makes it hard to predict the overall sieving time: for instance, even factoring times of numbers of comparable size can easily vary by 10 %. In this chapter we present a method for predicting the number of relations needed for factoring a given number in practice within 2 % of the actual number of relations needed. With 'in practice' we mean: on a given computer, for a given implementation, and for a given choice of the parameters in the NFS. This allows us to predict the actually required sieving time within 2 %. Our method is based on a short sieving test and a very cheap simulation of the relations needed for the factorization. By applying this method for various choices of the parameters of the number field sieve, it is possible to find good parameters, e.g., in terms of minimizing the sieving time or the size of the resulting matrix.

Our method works as follows. After choosing polynomials, a factorbase bound F and a large prime bound L (for ease of exposition we take the same bounds on both the rational side and the algebraic side), and a sieve area, we perform a sieving test for a relatively short period of time. E.g., for a 120-digit number one could sieve for about ten minutes. Each relation consists of a pair of coprime integers (a, b) along with the corresponding factorizations of the rational and the algebraic norms. Together the rational factorizations found during the sieving test determine a certain distribution of the large primes occurring in them and the same is the case for the algebraic factorizations, though the distributions may be different. Since the number of relations to be generated depends on the matching behavior of the large primes, we randomly generate pairs of rational and algebraic large primes combinations according to the relevant distributions as observed in the sample (obviously not paying any attention to a pair (a, b), to which a pair of combinations would correspond, since such a pair (a, b) will in general not exist), and hope that the matching behavior of these cheaply generated *simulated relations* corresponds to the matching behavior of the actual relations. Thus, during the simulation we regularly count the number of simulated relations after singleton removal and assume that the point where the number of simulated relations would start to generate an oversquare matrix is a good estimate for the number of actual relations that will be required.

The details of this method are described in Section 4.2, where we pay special attention to the difference between line sieving and lattice sieving, as lattice sieving uses so-called special primes. However, we also notice similar behavior of the distribution of the large primes in line and lattice sieving, if F and L are chosen equal. Here we distinguish two cases; if F and L are relatively close we consider it as Case I, else we consider it as Case II. This is also shown in Section 4.2. The line sieving data sets were generated with the NFS software package of CWI. The lattice sieving data sets were either given by Bruce Dodson and Thorsten Kleinjung or generated with Kleinjung's lattice siever.

Section 4.3 deals with the singleton removal and the estimate for the number of relations needed to factor the given number. For lattice sieving, the number of dupli-

cates has to be estimated as well. In Section 4.4 we compare results of the simulation with original factorizations and in Section 4.5 we describe in detail how to distinguish between Case I and Case II.

The sieving in NFS, combined with the simulation, has more interesting aspects, of which some are treated in Section 4.6. In Subsection 4.6.1 we start with determining the size of the sample sieve test, based on Chebyshev's inequality. In Subsection 4.6.2 we explain how to obtain an appropriate size of the sieving area and the corresponding sieving time by using our method for simulating the sieving step.

Another aspect concerns the growth behavior of the number of useful relations as a function of the total number of relations. Dodson and Lenstra describe the growth of useful relations as explosive [17], but we show in Subsection 4.6.3 that a more gradual growth is also possible. The final aspect we consider, in Subsection 4.6.4, is the relation between the oversquareness of a set of relations and the resulting matrix size.

4.2 Simulating relations

Before we start with the simulation, we run a short sieving test. In order to get a representative selection of the actual relations, we ensure that the points we are sieving in this test form a representative sample of the entire sieving area. The parameters for the sieving are set in such a way that we have at most two large primes both on the rational side and on the algebraic side. In the case of lattice sieving we have one additional special prime on one of the sides. In this section we describe the process of simulating relations both for line sieving and for lattice sieving. Note that we only simulate the large primes; for the primes in the factorbase we use a correction as will be explained in Section 4.3.

The first step after the sieving test consists of splitting the relations according to the number of large primes occurring in the relation. The set of relations with i large primes on the rational side and j large primes on the algebraic side is denoted by $r_i a_j$ for $i, j \in \{0, 1, 2\}$. This leads to nine different sets and the mutual ratios of their cardinalities determine the ratios by which we will simulate the relations. In the case of lattice sieving we split the relations in the same way, treating the special prime separately.

Our first experiments with simulating the large primes for the set $r_1 a_0$ (and removing singletons) concentrated on the large primes at hand. We tried linear interpolation between two consecutive large primes, Lagrange polynomials, and splines, but all these local approaches did not give a satisfying result: the number of relations after singleton removal was too far from the original data. We then tried a more global approach, looking at all the large primes and seeing if we could find a distribution for them. We found in this case that an exponential distribution simulates best the distribution of these large primes over the interval $[F, L]$ (cf. [7], Ch. 6) and the result after singleton removal was satisfying. The inverse of this distribution function

is given by

$$g(x) = F - a \log\left(1 - x\left(1 - e^{\frac{F-L}{a}}\right)\right), 0 \leq x \leq 1, \tag{4.1}$$

where a is the average of the large primes in the set $r_1 a_0$. Note that $g(0) = F$ and $g(1) = L$. In order to generate primes according to the actual distribution of the large primes, we generate a random number between 0 and 1, substitute this number in $g(x)$, round the number $g(x)$ to the nearest prime, and repeat this for each prime that we want to generate.

To avoid expensive prime tests, we work with the index of the primes p, defined as $i_p = \pi(p)$, rather than with the prime itself. This index can be found by using a look-up table or the approximation i_p being the nearest integer to $\frac{p}{\log p} + \frac{p}{\log^2 p} + \frac{2p}{\log^3 p}$ [37]. Experiments showed that this third order approximation gives almost the same results as looking up indices in a table. It is more efficient to use the approximation when L is large.

We choose for the simulation of the indices the same type of exponential distribution. This may seem strange, as prime numbers have a different distribution, but experiments showed that this choice gave good results after singleton removal. So we choose for simulating the indices of the large primes in the set $r_1 a_0$

$$G(x) = i_F - a' \log\left(1 - x\left(1 - e^{\frac{i_F - i_L}{a'}}\right)\right), 0 \leq x \leq 1, \tag{4.2}$$

where i_F stands for the index of the first prime above F, i_L for the index of the prime just below L, and a' for the average of the indices of the large primes in the set $r_1 a_0$.

During our experiments with various choices of functions for simulating the distributions of the large primes in the different sets of relations $r_i a_j$, we found that it is convenient to distinguish between two cases: the case in which the ratio F/L is approaching 1 (Case I) and the case in which this ratio is approaching 0 (Case II). These two cases ask for different choices of the distribution functions (described below). Table 4.18 in Section 4.5 gives an overview of the experiments which we have carried out for each case to illustrate our method for simulating relations.

All functions that we give for simulating relations in the following two subsections were found experimentally. We have no theoretical proof of why the distributions of the large primes found during the sieving stage follow these distributions, only experimental proof that the result after removing singletons of the simulated relations, generated with these functions, is good.

4.2.1 Case I

If F and L are relatively close, we use the following approach for the different types of relations.

$\underline{r_0 a_0}$: We count the number of relations in this set.

$\underline{r_1 a_0}$: As mentioned before, for simulating the indices of the large primes, we use the function:

$$G_{\mathrm{I}}(x) = i_F - a' \log\left(1 - x\left(1 - e^{\frac{i_F - i_L}{a'}}\right)\right), 0 \leq x \leq 1, \tag{4.3}$$

with i_F and i_L the corresponding indices of F and L, respectively, and a' the average of the indices of the large primes in the set.

To illustrate that the distribution of the large primes is approximated well by (4.3) we have generated the following graph (Figure 4.1), which consists of two sorted sets. One set consists of the indices of the primes of the original sieving data and the other set consists of the indices simulated according to (4.3). The line of the simulated data is the one which lies below the other line (of the original data) around position 7000.

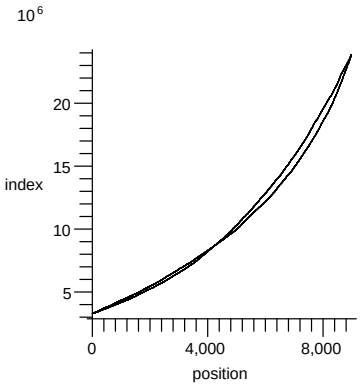

Figure 4.1: Comparing original and simulated data (r_1a_0)

$\underline{r_0a_1}$: We would like to use the same idea as we used for r_1a_0, but now we have to deal with algebraic primes. This means that not all primes can occur, and that each prime that does occur can have up to d different roots, where d is the degree of the polynomial $f_2(x)$. This yields pairs of a prime and a root which we denote by $(prime, root)$. Luckily, (heuristically) the amount of pairs $(prime, root)$ with $F < prime < L$ is about equal to the amount of primes between F and L. This implies that we do not have to simulate pairs with a certain subset of indices, as we may assume that all indices can occur in the simulation. We found that an exponential distribution fits here as well, so here we use the same approach as we did for r_1a_0.
$\underline{r_1a_1}$: We assume that the value of the index on the rational side is independent of the value of the index on the algebraic side. So we combine the approach for r_1a_0 and that for r_0a_1. Using (4.3), generate a random number and compute the corresponding rational index, generate a new random number (do not use the first random number as input for the random number generator) and compute the corresponding algebraic index.
$\underline{r_2a_0}$: Here we have to deal with two large primes on the rational side, denoted by q_1 and q_2 with $q_1 > q_2$. We started with sorting the list with q_1 and we found that a

linear distribution fits these data well. This first surprised us, as we expected a type of exponential distribution. However, if we look at the first terms of the Taylor series of (4.3) and take $\frac{i_F - i_L}{a'}$ as an independent variable, we get

$$i_F + a' \left(\left(\frac{i_F - i_L}{a'} \right) x + \frac{1}{2} \left(\frac{i_F - i_L}{a'} \right)^2 x \right).$$

If i_F and i_L are relatively close, the first term of the series determines the output. So the distribution function of the index i_{q_1} of q_1 is given by

$$H_{\mathrm{I},1}(x) = i_F + x(i_L - i_F),$$

where x is a number between 0 and 1.

We continued with q_2 and sorted them. Here, an exponential distribution fits the data, but we have to take into account that $q_2 < q_1$. Remember that we need an average value for the exponential distribution, but we cannot use all q_2-values. Instead of using one average value, we make a list of averages a_{q_2} of the sorted q_2-indices. We denote the average of the first j q_2-indices by $a_{q_2}[j]$ $(j = 1, 2, \ldots)$.

Now we describe how to simulate elements of $r_2 a_0$. We begin with a random number between 0 and 1 and compute $H_{\mathrm{I},1}(x)$, which gives us an index i_{q_1} of q_1. We look up this index in the sorted list of q_2-indices and the corresponding position j tells us which average we should use for computing the index i_{q_2} of q_2. We generate a new random number between 0 and 1 and substitute it for x in the following formula for $H_{\mathrm{I},2}(x)$, which is an adjusted form of $G_{\mathrm{I}}(x)$:

$$H_{\mathrm{I},2}(x) = i_F - a_{q_2}[j] \log \left(1 - x \left(1 - e^{\frac{i_F - i_{q_1}}{a_{q_2}[j]}} \right) \right).$$

This formula gives us an index i_{q_2} of q_2 that is smaller than the index we generated for q_1.

We made two graphs (cf. Figure 4.2) similar to Figure 4.1, but for q_1 (graph on the left-hand side) and q_2 (graph on the right-hand side). Especially the simulation of q_2 is very close; for q_1 the gap is larger (in the graph on the left-hand side the simulation is given by the upper line around position 60 000, in the graph on the right-hand side the simulation is given by the lower line around position 90 000). It might be possible to find an even better approximation of q_1, but as the result after singleton removal is very good, it will not be worth the effort from a practical point of view.

Given the distributions of the largest and second largest prime, we wondered how the products of the two primes are distributed. To illustrate this, we took the data set of $13,220+$ (cf. Subsection 4.4.1) found by our implementation of the siever. We added for each relation in $r_2 a_0$ the indices of the two large primes and split the interval $[2i_F, 2i_L]$ in ten equal subintervals (labeled $s = 1, \ldots, 10$). For each subinterval we counted the number of relations for which the sum of the two indices of the two large primes lies in this subinterval. The result is given in Table 4.1.

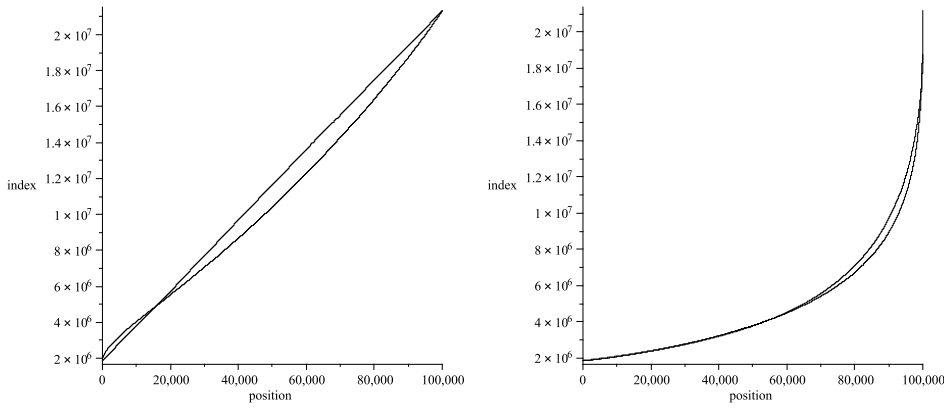

Figure 4.2: Comparing original and simulated data (q_1 left, q_2 right)

Table 4.1: Distribution of the sum of the indices $(13,220+)$

s	# relations
1	120780
2	161735
3	148757
4	133845
5	121967
6	78725
7	39253
8	20710
9	8107
10	0

This is consistent with the smallest prime having an exponential distribution. The zero in the last column is due to one of the bounds in the siever, which was set at $F^{0.1}L^{1.9}$ instead of L^2.

$\underline{r_0a_2}$: We know how to deal with r_2a_0 and we apply the same approach to r_0a_2, as we can make the same transition as we made from r_1a_0 to r_0a_1.

Sorting the list with q_1 showed that we could indeed use a linear distribution and the sorted list with q_2 showed that an exponential distribution fitted here. Now we simulate elements of r_0a_2 in the same way as elements of r_2a_0.

$\underline{r_1a_2}$: As with r_1a_1, we assume that the rational side and the algebraic side are independent. Here we combine the approaches of r_1a_0 and r_0a_2 to get the elements of r_1a_2.

$\underline{r_2a_1}$: Combine the approaches of r_2a_0 and r_0a_1 to get the elements of r_2a_1.

$\underline{r_2a_2}$: We combine the approaches of r_2a_0 and r_0a_2.

Summarizing, Case I is based upon the following assumptions:

- i_F and i_L are relatively close,
- the probability that the rational side is (semi)smooth is independent of the probability that the algebraic side is (semi)smooth for a pair (a, b) in the sieving area (as a consequence, the simulated value of a rational prime has no influence on the value of an algebraic prime),
- if the sieving bounds are the same on both sides, the rational prime(s) and algebraic prime(s) follow the same distribution and are simulated with the same model,
- the above model for one large prime ($G_{\mathrm{I}}(x)$, described in r_1a_0),
- the above model for two large primes ($H_{\mathrm{I},1}(x)$ and $H_{\mathrm{I},2}(x)$, described in r_2a_0).

4.2.2 Case II

We now look at data sets with F and L far apart and describe which functions simulate best the distribution of the indices of the large primes in these sets. The basic idea is the same as in Case I; we only use slightly different approximating functions.

First, we give the model for one large prime. For ease of exposition, we show it for the set r_1a_0. In Case I we use $G_{\mathrm{I}}(x)$ to simulate the indices of this set, but this function does not fit any longer. This is due to the behavior of $e^{(i_F-i_L)/a'}$ when i_L is much larger than i_F. After some experiments we decided to fit the following function, given the relations of the sieving test:

$$G_{\mathrm{II}}(x) = e^{(\alpha x+\beta)^\gamma}, 0 < \gamma \leq 1, 0 \leq x \leq 1, \tag{4.4}$$

with $\alpha = (\log(i_L))^{1/\gamma} - (\log(i_F))^{1/\gamma}$ and $\beta = (\log(i_F))^{1/\gamma}$, so we have $G_{\mathrm{II}}(0) = i_F$ and $G_{\mathrm{II}}(1) = i_L$. To determine γ, we apply the least squares method. Suppose we have n relations in the set r_1a_0, and the indices of the large primes are sorted into an array lp with $lp[0] \leq lp[1] \leq \ldots \leq lp[n-1]$. Then let

$$S := \sum_{i=1}^{n} \left(lp[i-1] - G_{\mathrm{II}}\left(\frac{i}{n+1}\right)\right)^2.$$

Select the γ for which the sum S has the lowest value by either trying all γ from 0.01 to 0.99 with step 0.01 or by using a more sophisticated method like Newton's method.

As an example, we show for B449 (a number of 449 bits) a graph with the original indices and a graph with the simulated indices of r_1a_0 (Figure 4.3). The indices are sorted and the place of the index in this sorted list is indicated by position. We have chosen $F = 4\mathrm{M}$ and $L = 2^{30}$ ($\approx 1074\mathrm{M}$). It turned out that $\gamma = 0.44$ fitted best after trying all values for γ, for a sieving test of 0.11 % of the original data.

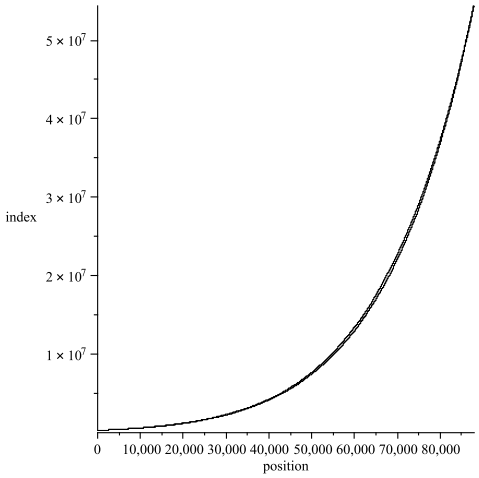

Figure 4.3: Comparing original and simulated data ($r_1 a_0$)

For two large primes we use the exponential distribution for the largest prime as well. We define

$$H_{\mathrm{II},1}(x) = i_F - a' \log\left(1 - x\left(1 - e^{\frac{i_F - i_L}{a'}}\right)\right), 0 \le x \le 1, \tag{4.5}$$

where a' is the average of the indices. For the second prime, the situation remains the same as in Case I. We make a list of averages a_{q_2} of the sorted q_2-indices, where $a_{q_2}[j]$ contains the average of the first j q_2-indices. After computing i_{q_1} with $H_{\mathrm{II},1}(x)$, we look up the corresponding average $a_{q_2}[j]$ and compute i_{q_2} with

$$H_{\mathrm{II},2}(x) = i_F - a_{q_2}[j] \log\left(1 - x\left(1 - e^{\frac{i_F - i_{q_1}}{a_{q_2}[j]}}\right)\right), 0 \le x \le 1.$$

To illustrate this, we show in Figure 4.4 for B449 the curves of indices for the original data and for the simulated data for the set $r_2 a_0$. Recall that we have $F = 4\mathrm{M}$ and $L = 2^{30} \approx 1074\mathrm{M}$.

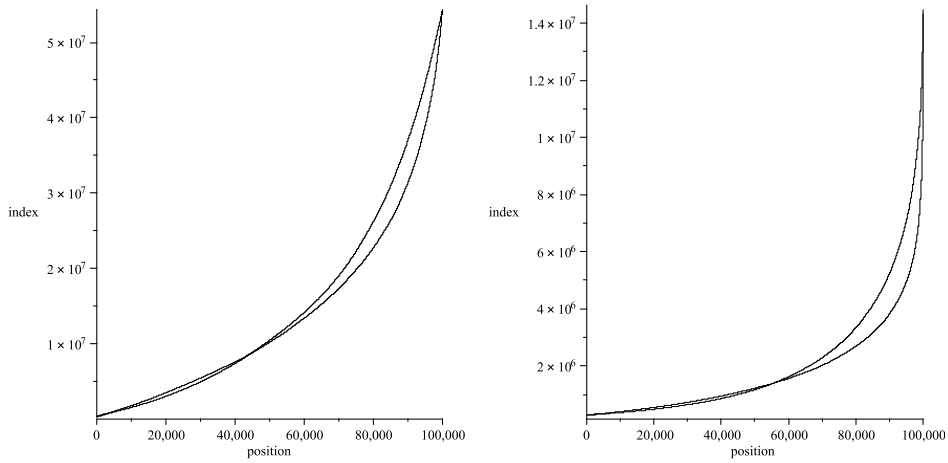

Figure 4.4: Comparing original and simulated data $r_2 a_0$ (q_1 left, q_2 right)

In both graphs the simulation is given by the lower line around position 90 000. As q_2 should be smaller than q_1, the difference in the second graph is a consequence of the difference in the first graph. As the approximating functions for the indices of the large primes are tailor-made for this data set, more data sets with similar sieving bounds are needed to test how widely applicable Case II is.

Summarizing, Case II is based upon the following assumptions:

- i_F and i_L are far apart,
- the probability that the rational side is (semi)smooth is independent of the probability that the algebraic side is (semi)smooth for a pair (a, b) in the sieving area (as a consequence, the simulated value of a rational prime has no influence on the value of an algebraic prime),
- if the sieving bounds are the same on both sides, the rational prime(s) and algebraic prime(s) follow the same distribution and are simulated with the same model,
- the above model for one large prime ($G_{\mathrm{II}}(x)$),
- the above model for two large primes ($H_{\mathrm{II},1}(x)$ and $H_{\mathrm{II},2}(x)$).

4.2.3 Special primes

In case of lattice sieving, we simulate the relations as described in either Case I or Case II and add a special prime to each relation in the following way. From the relations in the sieving test we compute the average number of relations per pair $(\mathit{special\ prime}, \mathit{root})$. Then we divide the number of relations we want to simulate

by this average and this gives the total number of special primes in our simulation. We select an appropriate interval for the special primes from which they are chosen as follows. Divide the interval in a (small) number of sections: per section select randomly the special primes and add one special prime to each of the relations. By dividing in sections (and simulating the same amount of relations per section) we make sure that each part of the interval of special primes is well presented, and by choosing randomly in each section, we get enough variation in the amount of relations per special prime. If the interval of the special primes is very large, it might become necessary to decrease the number of relations per section. A sieving test, of which the special primes are uniformly distributed over the interval, will indicate a suitable choice of sections and the number of relations for each section.

It is possible to use different factorbase bounds for the rational primes and the algebraic primes, bound the product of the two large primes on the same side, etc. All these details in the sieving influence the relations, but once the general model is known, it is relatively easy to adjust it to match the details.

4.3 The stop criterion

We now know how to simulate relations, but how many should we simulate?

In order to factor the number N we have to find dependencies in a matrix, as explained in Section 4.1. For our stop criterion it is enough to know when we have enough relations, i.e. *when the number of relations after singleton removal exceeds the number of different primes that occur in the remaining relations.*

After the singleton removal, we count how many relations are left and how many different large primes occur in these relations. We define the percentage oversquareness O_r after singleton removal (s.r.) as

$$O_r := \frac{n_r}{n_l + n_F - n_f} \times 100,$$

where n_r is the number of relations after singleton removal, n_l is the number of distinct large primes after singleton removal, n_F is the number of primes in the factorbase, approximated by $\pi(F_{\text{rat}}) + \pi(F_{\text{alg}})$, and n_f is the number of free relations from factorbase elements. We have (cf. [18], Ch. 3):

$$n_f = \frac{1}{g}\pi(\min(F_{\text{rat}}, F_{\text{alg}})),$$

where g is the order of the Galois group of $f_1(x)f_2(x)$, and F_{rat} and F_{alg} refer to the rational and algebraic factorbase bound, respectively. As we use the free relations in a later stage, we could have chosen to add n_f in the numerator, instead of subtracting n_f in the denumerator. However, as n_f is relatively small, this only makes a small difference and if $O_r = 100$, both fractions are the same. If $O_r \geq 100$ we expect to find a dependency in the matrix, and we stop with simulating relations. To make

practically sure to find a dependency, we stop when $O_r = 102$. An even larger percentage is advised if one would like to have more choices in the relations that can form a dependency and subsequently form a smaller matrix in the linear algebra step.

4.3.1 Duplicates

One final point concerns lattice sieving. It is well known that lattice sieving produces lots of duplicates, especially when it involves many special primes. For instance, the authors of [3] report 16.58 % duplicates. We treat our relations as if there are no duplicates, but that implies that in the case of lattice sieving we have to add a certain number of relations to the relations that we should collect in the sieving stage. This number can be computed as in [3]. The basic idea in [3] is to run a sieving test, in which only a small fraction of the special primes is processed. These special primes are uniformly distributed over the special primes interval. For each relation found in this sieving test, the authors of [3] check if it has more than one prime in the special primes interval. If so, they check for each special prime in this relation if the relation is in the sieving region of the corresponding lattice and if the cofactor bounds are kept. If both answers are positive, this gives rise to a duplicate relation. By processing all relations in the sieving test in the above way, we get the number of expected duplicates for this fraction of the special primes. Thus we also know the number of expected duplicates for the entire interval.

4.4 Experiments

We have applied our method to several original data sets (coming from factored numbers) and show that this gives good results. We have carried out two types of experiments.

First we assumed that the complete data set is given and we wanted to know if the simulation gave the same oversquareness when simulating the same number of relations as contained in the original data set. As input for the simulation we used 0.1 % of the original data (taking 1 out of every 1000 relations).

Secondly we assumed that only a small percentage (0.1 %) of the original data is known. Based on this data we simulated relations until $O_r \geq 100$. Then we compared this with the oversquareness of the same number of original relations.

This 0.1 % was found in an empirical way. We started a simulation based on 100 % original data and lowered this percentage in the next experiment until results after singleton removal were too far from the original data. We went down as far as 0.01 %, but this percentage did not always give good results, unless we would have been satisfied with an estimate within 5 % of the original data (although some experiments with 0.01 % of the original data were even as good as the ones based on 0.1 % of the real data). Another approach, based on Chebyshev's inequality, is given in Subsection 4.6.1.

4.4.1 Case I, line sieving

Some relevant parameters for all the original data sets in this section are given in Table 4.2, where M stands for million. Numbers are written in the format $a, b+$ or $a, b-$, meaning $a^b + 1$ or $a^b - 1$. In the case of GNFS, some prime factors were already known and for the remaining factors it was more efficient to use GNFS instead of SNFS.

Table 4.2: Sieving parameters (line sieving)

number	# dec. digits	F	L	g	$n_F - n_f$
13,220+	117	30M	400M	120	3700941
26,142+	124	30M	250M	120	3700941
19,183−	131	30M	250M	18	3613192
66,129+	136	35M	300M	18	4175312
80,123−	150	55M	450M	18	6383294

The experiments for the first two GNFS data sets $13,220+$ and $26,142+$ are in Table 4.3. Here, O stands for the original data and S for the simulated data. Table 4.3 shows that the numbers were oversieved, but the simulated data show about the same oversquareness. In Table 4.4, we computed the relative difference (S−O)/O $\times$ 100 of the entries in the S- and O-column of Table 4.3. We see that our predictions of the number of relations after singleton removal (n_r), the number of large primes after singleton removal (n_l), and the oversquareness (O_r) differ from the original data by not much more than 1 %.

Table 4.3: Experiments line sieving

GNFS	13,220+ O	13,220+ S
# relations before s.r.	35 496 483	35 496 483
n_r	21 320 864	21 394 640
n_l	13 781 518	13 950 420
O_r (%)	121.96	121.21
	26,142+ O	26,142+ S
# relations before s.r.	23 580 294	23 580 294
n_r	15 150 790	15 253 825
n_l	9 448 082	9 397 751
O_r (%)	115.22	116.45

Table 4.4: Relative differences of Table 4.3 results

GNFS	13,220+	26,142+
n_r (%)	0.35	0.68
n_l (%)	1.22	−0.53
O_r (%)	−0.61	1.07

We give the following timings for these experiments: simulation of the relations, singleton removal, and actual sieving time (Table 4.5). The simulations in this section and the rest of the chapter were carried out on an Intel® Core™2 Duo desktop with 2 GB of memory. For the actual sieving we used multiple machines and added the sieving times of each machine. As we used 0.1 % data, we have to keep in mind that we need to add 0.1 % of the sieving time to the time of a complete experiment, which consists of generating a small data set, simulate a big data set, and remove singletons. When we change parameters in NFS we have to generate a new data set.

Roughly speaking, we can say that the calculation of the prediction of the total sieving time (for a given choice of the NFS parameters) by our method costs less than one CPU hour, whereas the actual sieving costs several hundreds of CPU hours.

Table 4.5: Timings

GNFS	13,220+	26,142+
simulation (sec.)	224	156
singleton removal (sec.)	927	573
actual sieving (hrs.)	316	709

For our second type of experiments, we assume that we only have a small sieving test of the number to be factored. When are we in the neighbourhood of $O_r = 100$ according to our simulation and will the original data agree with our simulation? We started to simulate 5M, 10M, ... relations (with increment 5M) and for these numbers we computed the oversquareness O_r; when O_r approached the 100 bound we decreased the increment to 1M. Table 4.6 gives the number of relations for which O_r is closest to 100 and the next O_r (for 1M more relations), both for the simulated data and the original data. This may of course be refined.

Table 4.6: Around 100 % oversquareness (GNFS)

# rel. before s.r.	O_r S (%)	O_r O (%)	relative diff. (%)
28M (13,220+)	99.66	99.87	−0.21
29M (13,220+)	103.15	103.29	−0.14
20M (26,142+)	100.57	99.24	1.34
21M (26,142+)	105.38	104.03	1.30

For SNFS the higher degree polynomial has small coefficients, which leads to a shorter sieving step. Tables 4.7–4.10 show the same kind of data as Tables 4.3–4.6, but now for SNFS. We start in Table 4.7 with the complete data set and simulate the same number of relations. Table 4.8 gives the relative differences of the results of the experiments in Table 4.7. The timings are given in Table 4.9.

Table 4.7: Experiments line sieving

SNFS	# rel. before s.r.	n_r	n_l	O_r (%)
19,183− O	21 259 569	11 887 312	7 849 531	103.70
19,183− S	21 259 569	12 156 537	7 936 726	105.25
66,129+ O	26 226 688	15 377 495	10 036 942	108.20
66,129+ S	26 226 688	15 656 253	10 123 695	109.49
80,123− O	36 552 655	20 288 292	12 810 641	105.70
80,123− S	36 552 655	20 648 909	12 973 952	106.67

Table 4.8: Relative differences of Table 4.7 results

SNFS	19,183−	66,129+	80,123−
n_r (%)	2.26	1.81	1.78
n_l (%)	1.11	0.86	1.27
O_r (%)	1.49	1.19	0.92

Table 4.9: Timings

SNFS	19,183−	66,129+	80,123−
simulation (sec.)	128	166	223
singleton removal (sec.)	487	603	771
actual sieving (hrs.)	154	197	200

In Table 4.10 we simulate the number of relations that leads to an oversquareness around 100 %. We compare this number with the original data and give the differences in oversquareness.

Table 4.10: Around 100 % oversquareness (SNFS)

# rel. before s.r.	O_r S (%)	O_r O (%)	relative diff. (%)
20M (19,183−)	99.22	97.71	1.55
21M (19,183−)	104.06	102.51	1.51
23M (66,129+)	96.44	95.35	1.14
24M (66,129+)	100.72	99.60	1.12
34M (80,123−)	99.93	98.66	1.29
35M (80,123−)	102.82	101.50	1.30

All these data sets were generated with the NFS software package of CWI, and the models for describing the underlying distributions were the same for SNFS and GNFS, as described in Section 4.2.

4.4.2 Case I, lattice sieving

For lattice sieving we used a data set from Bruce Dodson (7,333−, SNFS). Besides the factorbase bound and the large primes bound, we have two intervals for the special primes. These are given in Table 4.11.

Table 4.11: Sieving parameters (lattice sieving)

	7,333−
# dec. digits	177
F	16 777 215
L	250 000 000
ranges of special primes	[16 777 333, 29 120 617]
	[60 000 013, 73 747 441]
g	6
$n_F - n_f$	1 976 740

As we are now dealing with lattice sieving, we have an extra (special) prime to simulate, in the way described in Subsection 4.2.3. Fortunately, the distribution of the other large primes is similar to that for line sieving. The results of our experiments are given in Table 4.12, based on 0.023 % of the original data. The last line in this table is the total number of relations without duplicates. In total 26 024 921 relations were found after completion of the sieving.

Table 4.12: Oversquareness 7,333−

# rel. before s.r.	O_r S (%)	O_r O (%)	relative diff. (%)
17M (7,333−)	98.34	97.45	0.91
18M (7,333−)	103.96	103.08	0.85
25 112 543 (7,333−)	135.39	136.64	−0.91

4.4.3 Case II, line sieving

As an example of Case II for line sieving, we show the experiments of B449. The parameters used for the sieving are given in Table 4.13.

- B449 = 124485 3679600401 0445323086 2441169510 2215499569 1394018820 4269680737 8014739981 2209253057 0913282691 2163486784 5908830265 1464632317 1300493001

Table 4.13: Sieving parameters B449 (line / lattice sieving)

	B449
# dec. digits	136
F_{rat}	4 000 000
F_{alg}	9 000 000
L	2^{30}
upper bound on the product of two large primes	2^{56}
ranges of special primes	[9 000 000, 17 000 000] [19 000 000, 22 897 969]
g	120
$n_F - n_f$	883 275

We show the graphs for the original and simulated data for $r_1 a_0$ and $r_2 a_0$ in Figures 4.5 and 4.6, respectively. The simulation lines in Figure 4.6 are the lower lines around position 90 000.

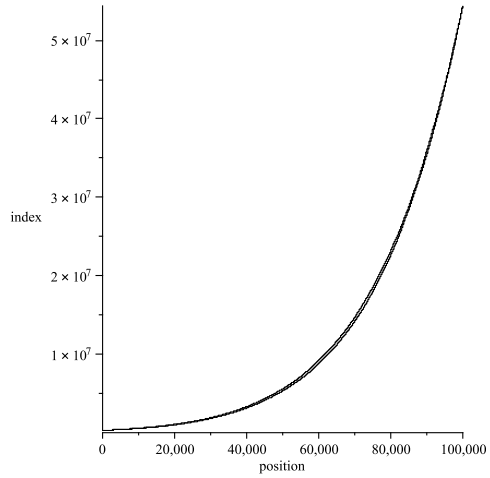

Figure 4.5: Comparing original and simulated data $r_1 a_0$

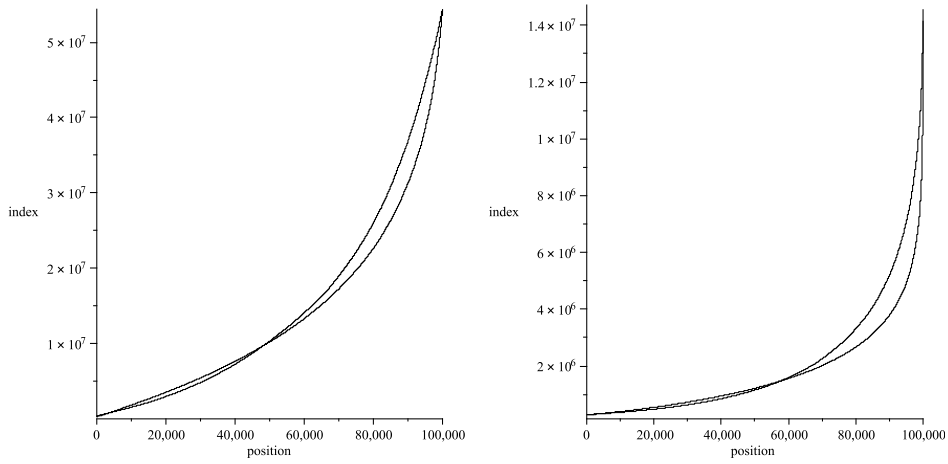

Figure 4.6: Comparing original and simulated data $r_2 a_0$ (q_1 left, q_2 right)

The results of simulating the line sieving data set are in Table 4.14. The simulation is based on 0.1 % of the original data set.

Table 4.14: Oversquareness B449 (line sieving)

# rel. before s.r.	O_r S (%)	O_r O (%)	relative diff. (%)
40M	99.99	96.02	4.13
41M	102.11	98.37	3.80
44 444 780	109.32	105.57	3.55

Due to the absence of special primes, we reach the 100 % oversquareness a bit later than with lattice sieving (cf. Subsection 4.4.4).

4.4.4 Case II, lattice sieving

We also used lattice sieving on the same B449. The parameters used for the sieving, including the interval of the special primes, are given in Table 4.13. As before, we simulate the complete original data set and we estimate when we have reached 100 % oversquareness. The results of our experiments are given in Table 4.15, based on 0.11 % original data. The last line in this table is the total number of relations without duplicates. In total 48 885 461 relations were found after completion of the sieving.

Table 4.15: Oversquareness B449 (lattice sieving)

# rel. before s.r.	O_r S (%)	O_r O (%)	relative diff. (%)
37M	98.80	88.41	11.75
38M	101.43	95.00	6.76
44 401 665	115.02	107.20	7.29

We already saw in Subsection 4.2.2 that the model for two large primes had both lines of q_1 and q_2 below the original data. The consequence is that the simulated data has more primes in a smaller interval, and thus reaches the 100 % oversquareness earlier than the original data. The effect is even stronger as the special primes are also in this interval.

4.4.5 Comparing line and lattice sieving

In the two subsections on Case II examples, we used the same sieving bounds for generating the data set for line sieving and for lattice sieving. In the present subsection we choose again the same bounds, but now with F and L closer to each other so that we get a Case I example. We performed the sieving for the same number $13,220+$ (cf. Subsection 4.4.1), but now with a different large prime bound. We used $F = 30\text{M}$

and $L = 2^{29} \approx 537\text{M}$, and the special primes were chosen in the interval $[30\text{M}, 35\text{M}]$. We started with line sieving and for the simulation we used the models as described in Case I. The next two graphs (Figures 4.7 and 4.8) show the behavior of the large primes of $r_1 a_0$ and $r_2 a_0$. In Figure 4.7, the simulation is given by the lower line around position 90 000. In Figure 4.8, in the graph on the left-hand side the simulation of q_1 is given by the upper line around position 60 000 and in the graph on the right-hand side the simulation of q_2 is given by the lower line around position 90 000.

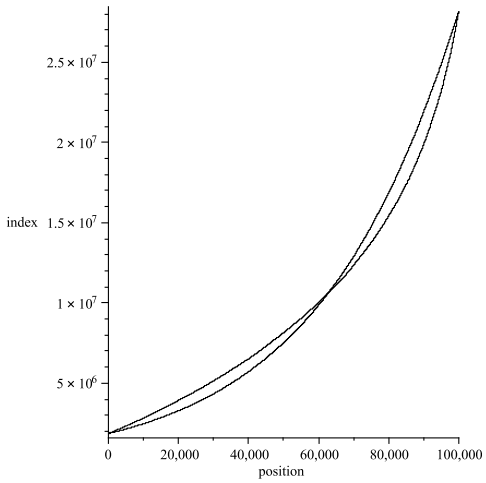

Figure 4.7: Comparing original and simulated data $r_1 a_0$

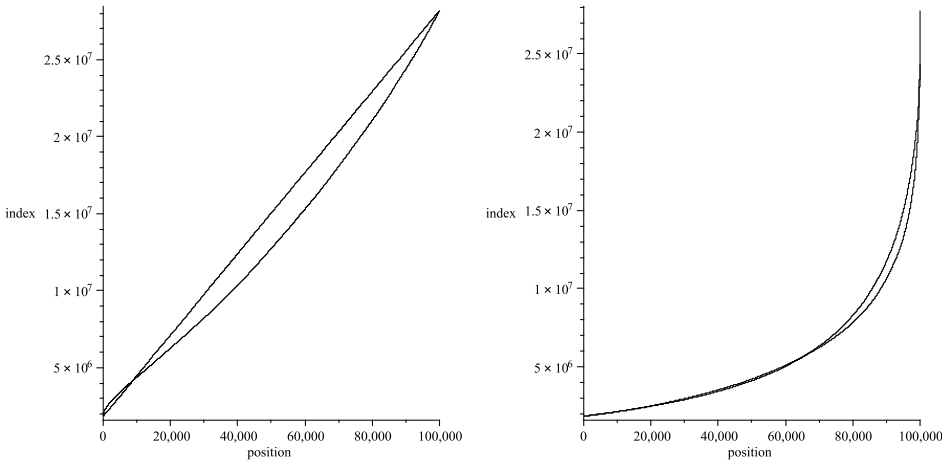

Figure 4.8: Comparing original and simulated data $r_2 a_0$ (q_1 left, q_2 right)

We give the oversquareness results of some experiments in Table 4.16. The simulations are based on 0.1 % of the original data.

Table 4.16: Oversquareness 13,220+ (line sieving)

# rel. before s.r.	O_r S (%)	O_r O (%)	relative diff. (%)
32M	97.50	100.75	−3.23
33M	100.61	103.65	−2.93
48 387 564	132.80	135.64	−2.09

To see if the models in Case I work as well for Kleinjung's lattice siever, we repeat the sieving for 13,220+ with Kleinjung's lattice siever with $F = 30\text{M}$ and $L = 2^{29} \approx 537\text{M}$ and the special primes between 30M and 35M. As before, we give the graphs of $r_1 a_0$ and $r_2 a_0$ (Figures 4.9 and 4.10). The lines of the simulated data are in the same position as in Figures 4.7 and 4.8.

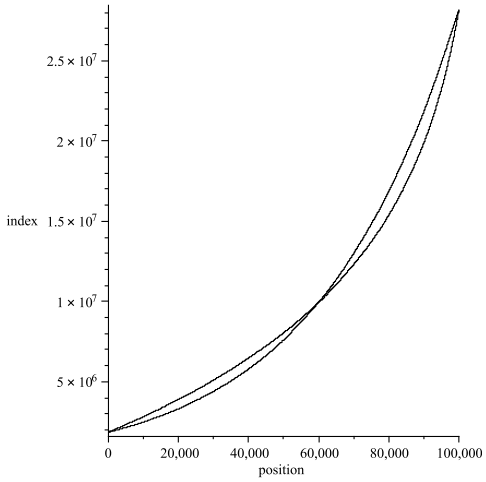

Figure 4.9: Comparing original and simulated data $r_1 a_0$

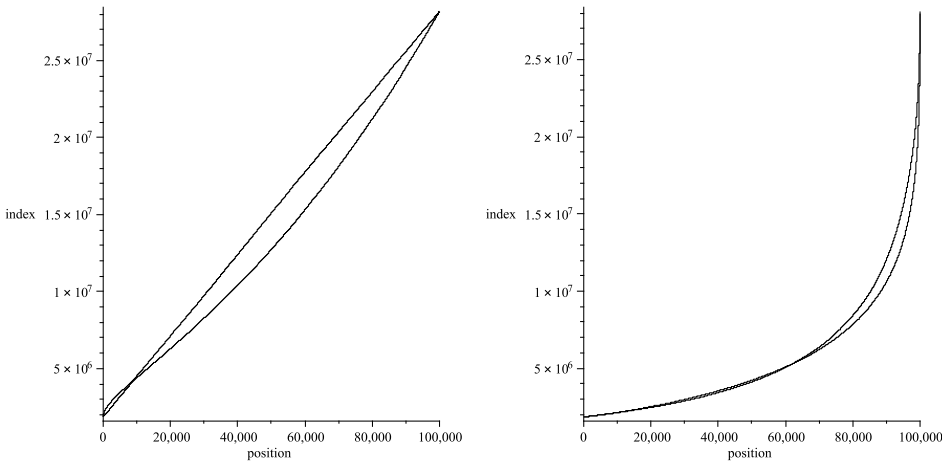

Figure 4.10: Comparing original and simulated data r_2a_0 (q_1 left, q_2 right)

The results, based on 0.1 % of the data, are given in Table 4.17. Note that the complete data set is smaller than the corresponding data set of line sieving, as we used a better estimate of the sieve area, required to factor the number.

Table 4.17: Oversquareness 13,220+ (lattice sieving)

# rel. before s.r.	O_r S (%)	O_r O (%)	relative diff. (%)
32M	98.45	100.17	−1.72
33M	101.51	103.15	−1.59
36 526 526	110.86	112.31	−1.29

Again, the graphs and oversquareness of the two original data sets show good resemblance.

Our observation that the same model can be used both for line sieving and lattice sieving having the same sieving bounds F and L may be explained as follows. In lattice sieving, the special primes are always chosen in an interval $[F, F']$, where $(F' - F)/(L - F)$ is small. For example, for $13,220+$ we have $(F' - F)/(L - F) = (35M - 30M)/(2^{29} - 30M) \approx 0.01$. This means that in lattice sieving the large primes occur in about the same interval as in line sieving.

4.5 Which case to choose

As we found that the models for simulating the large primes seem to depend largely on the chosen sieving bounds, we should be able to tell which model to use, given

a number to factor and the sieving parameters. We start with an overview of the experiments so far (Table 4.18), where we focus on the used sieving bounds and the corresponding model we used for the simulation. If there are two values for F, it should be read as F_{rat} / F_{alg}, else both sides use the same factorbase bound. For the convenience of the reader, we give as well the ratio of F and L as a percentage.

Table 4.18: Sieving bounds and corresponding case

number	F	L	L/F	line / lattice	Case
$13,220+$	30M	400M	13.33	line	I
$26,142+$	30M	250M	8.33	line	I
$19,183-$	30M	250M	8.33	line	I
$66,129+$	35M	300M	8.57	line	I
$80,123-$	55M	450M	8.18	line	I
$7,333-$	16.78M	250M	14.90	lattice	I
$13,220+$	30M	536.87M	17.90	line	I
$13,220+$	30M	536.87M	17.90	lattice	I
B449	4M / 9M	1073.74M	268.44 / 119.30	line	II
B449	4M / 9M	1073.74M	268.44 / 119.30	lattice	II
B454	5M / 10M	4294.97M	858.99 / 429.50	lattice	II

For the Case II experiments, we had a much smaller factorbase bound and a larger large prime bound than for the Case I experiments. To see the influence of taking only a larger large prime bound, we looked at the line sieve data set of another number, viz. $12,287+$ with $F_{\text{rat}} = 90$M, $F_{\text{alg}} = 300$M, and $L = 1070$M. If we look at the graphs (Figures 4.11 and 4.12) for r_1a_0, r_0a_1, r_2a_0, and r_0a_2, we see a behavior that resembles Case I. Note that we plot large primes of both r_2a_0 and r_0a_2 in one graph. In Figure 4.12, q_1 is given by the upper line and q_2 by the lower line.

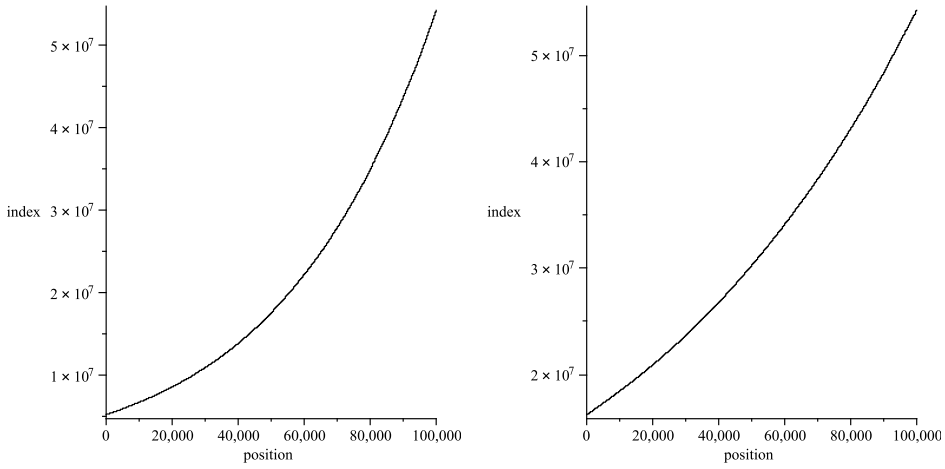

Figure 4.11: Original data of r_1a_0 and r_0a_1

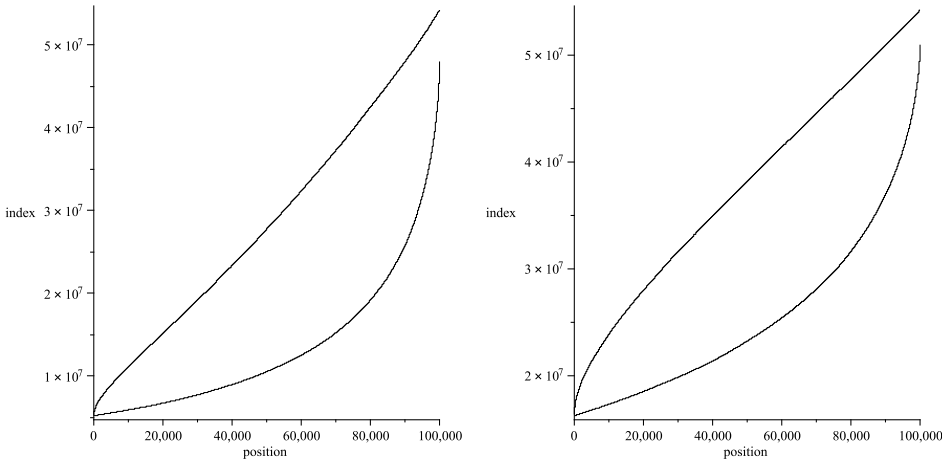

Figure 4.12: Original data of r_2a_0 and r_0a_2

The simulation of the complete line sieve data set confirms the resemblance of Case I, as we get almost the same oversquareness (Table 4.19).

Table 4.19: Experiment 12,287+ (line sieving)

GNFS	12,287+ O	12,287+ S
# relations before s.r.	91 159 660	91 159 660
n_r	58 261 730	58 794 418
n_l	32 318 386	32 284 786
O_r (%)	108.40	109.46

Table 4.18, combined with the simulation of the data set of $12,287+$ by using the models in Case I, suggests that varying the factorbase bound has a larger influence on which model to prefer than varying the large prime bound. Based on the experiments we did so far, a large prime bound that is at most 20 times the factorbase bound requires the models of Case I for the simulation, whereas a large prime bound that is at least 100 times the factorbase bound requires the models of Case II. By making a graph of the large primes of $r_1 a_0$ and $r_2 a_0$ in the sieving test, we decide which model to use.

4.6 Additional experiments

In Section 4.2 we have presented a method for predicting the sieving effort for the number field sieve. Here we look at some of the details of that method. In Subsection 4.6.1 we study the influence of the size of the sample sieving test on the average of the indices, hence on the quality of the simulation. We continue in Subsection 4.6.2 with determining the optimal size of the sieve area and computing the corresponding expected total sieving time.

We look in Subsection 4.6.3 at the number of useful relations compared with the total number of relations. Dodson and Lenstra describe the growth of useful relations as explosive [17], whereas we observe that a more gradual growth is also possible, if the larger factorbases required are feasible, and would lead to a faster overall factorization. We end this section with studying the influence of the amount of oversquareness on the resulting matrix size in Subsection 4.6.4.

4.6.1 Size of the sample sieve test

We found empirically that if we have 0.1 % of the data set available, our predictions of the sieving effort are satisfactory. However, a lower percentage sometimes worked as well. Recall that we used the small data set for (among others) computing the average of the indices. So we have to answer the following question: how large should a randomly chosen subset be in order that the average value of the subset is sufficiently close to the average of the complete set. For this we can use the so-called **law of large numbers** (cf. [16], Ch. 13):

Theorem 8 *If $\bar{X}_n$ is the average of n independent random variables with expectation μ and variance σ^2, then for any $\varepsilon > 0$:*

$$\lim_{n\to\infty} \mathrm{P}(|\bar{X}_n - \mu| > \varepsilon) = 0.$$

Here, P denotes the probability. Theorem 12 can be proved with the help of **Chebyshev's inequality**, which states:

for an arbitrary random variable Y and any $\varepsilon > 0$:

$$\mathrm{P}(|Y - \mathrm{E}[Y]| \geq \varepsilon) \leq \frac{1}{\varepsilon^2}\mathrm{Var}(Y).$$

As usual, $\mathrm{E}[Y]$ is the expectation of Y and $\mathrm{Var}(Y)$ is the variance of Y. Apply this inequality with $Y = \bar{X}_n$, $\mathrm{E}[\bar{X}_n] = \mu$ and $\mathrm{Var}(\bar{X}_n) = \sigma^2/n$. This leads to

$$\mathrm{P}(|\bar{X}_n - \mu| > \varepsilon) \leq \frac{\sigma^2}{n\varepsilon^2}. \tag{4.6}$$

By letting n tend to ∞, we get the law of large numbers. A consequence of the law of large numbers concerns the variance of a subset. We have with very high probability that $s_n^2 \to \sigma^2$ as $n \to \infty$, where $s_n^2 = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \bar{X}_n)^2$ is the variance of the subset [16]; s_n^2 is an unbiased estimator of σ^2, i.e., $\mathrm{E}[s_n^2] = \sigma^2$. If the variance of the complete data set is not known, we will use s_n^2, as this is the best possible estimation.

Depending on the extent to which we want to approximate the average we can compute n by using (4.6). To see which percentage deviation of the average is acceptable (which will be expressed by ε in (4.6)), we performed some experiments with a fixed average for r_1a_0 in case of $13,220+$ with $F = 30\mathrm{M}$ and $L = 400\mathrm{M}$. The complete data set has 2 501 147 relations in r_1a_0 with an average index of 8.161×10^6. After removing singletons we have 432 118 relations and 206 162 large primes, which has a ratio of 2.096. In the seven experiments referred to in Table 4.20 we generated 2 501 147 relations with the given average and the exponential distribution $G_{\mathrm{I}}(x)$,

$$G_{\mathrm{I}}(x) = i_F - a' \log\left(1 - x\left(1 - e^{\frac{i_F - i_L}{a'}}\right)\right);$$

subsequently we removed the singletons and computed the ratio. The first row refers to the simulation with the average of the original data set. The percentages given in the next rows indicate the deviation of the average. Here s.r. stands for singleton removal and l.p. for large primes.

Table 4.20: Average of r_1a_0 and simulation

average	# rel. after s.r.	# l.p. after s.r.	ratio
8.161×10^6	411 809	198 246	2.077
7.345×10^6 $(-10\,\%)$	433 180	207 862	2.084
7.753×10^6 $(-5\,\%)$	421 930	202 896	2.080
8.080×10^6 $(-1\,\%)$	412 307	198 436,	2.078
8.243×10^6 $(1\,\%)$	408 460	196 819	2.075
8.569×10^6 $(5\,\%)$	401 088	193 368	2.074
8.977×10^6 $(10\,\%)$	393 395	189 833	2.072

Due to the small deviation between the exponential model and original data, we see in Table 4.20 that the simulation with the average of the complete data set gives a deviation of almost 5 % in the number of relations and large primes, but the ratio stays within 1 % of the ratio of the original data. This indicates that a small deviation of the average has no major consequences for the oversquareness. The factorbase bound and the large prime bound are the same for all types of relations, and we expect the same behavior for the other types, since either they have the same exponential distribution or a simpler linear distribution. This is in compliance with our observation that our method for simulating relations is a robust method; the deviation has to be large in order to notice it. So we took this case to work out the details, as we expect the same behavior for other data sets.

We need to know the variance $\sigma^2 = \frac{1}{n}\sum_{1\le i\le n} x_i^2 - \mu^2$, in order to apply Chebyshev's inequality. For the complete set r_1a_0, we have $\sigma^2 \approx 2.938 \times 10^{13}$. If we want the average to be correct up to 5 % ($\varepsilon = 0.05\mu$) in 95 % of the cases (P = 0.05), we solve $0.05 = \frac{2.938\times 10^{13}}{(0.05\mu)^2 n}$, where $\mu = 8.161 \times 10^6$, which gives $n = 3529$. If we enlarge the deviation to 10 % of the average ($\varepsilon = 0.1\mu$, P = 0.05), we get $n = 882$. In the experiment with 0.1 % data referred to in Table 4.3, we have the following partitioning in the small data set from the sieving test (Table 4.21):

Table 4.21: Number of relations in sieving test

type	#
r_0a_1	5859
r_0a_2	5859
r_1a_0	2490
r_1a_1	7542
r_1a_2	7833
r_2a_0	837
r_2a_1	2560
r_2a_2	2535

We see that the smallest number in Table 4.21, 837, agrees with our computation $n = 882$ when we accept a deviation to 10 % of the average in 95 % of the cases. This indicates that the relation type that occurs the least is determining the size of the sieving test. It also implies that the other types will be better approximated. In case one type relation occurs much less frequently, we might accept an even larger deviation for this type, as this type relation might have a smaller influence on the oversquareness of the complete data set. However, this needs to be investigated further, because it might also be possible that although a certain type relation occurs rarely, it might still be important in the singleton removal phase.

In the computations above we used the average and variance of the complete data set, but these are unknown when we start with a new number. However, we can perform a small sieving test and use the average and variance of this test, assuming that these are representative for the complete data set. By applying Chebyshev's inequality, we get the minimum size of a good sieving test. We demonstrate this by repeating the computation for $13,220+$ for approximately 0.01 % of the data (by sieving 0.01 % of the lines). We computed for all types of relations the average, variance and the least number of relations that allows a deviation of the average up to 10 % in 95 % of the cases. In Tables 4.22–4.24, the numbers between brackets after the type of relation indicates the numbers of relations in the sieving test.

Table 4.22: Average

type	rational, p_1	rational, p_2	algebraic, p_1	algebraic, p_2
r_0a_1 (597)			8.566×10^6	
r_0a_2 (639)			1.110×10^7	5.048×10^6
r_1a_0 (232)	8.323×10^6			
r_1a_1 (770)	8.069×10^6		8.466×10^6	
r_1a_2 (743)	8.176×10^6		1.159×10^7	5.131×10^6
r_2a_0 (88)	1.017×10^7	4.553×10^6		
r_2a_1 (231)	1.089×10^7	5.346×10^6	7.823×10^6	
r_2a_2 (254)	1.090×10^7	4.963×10^6	1.132×10^7	5.232×10^6

Table 4.23: Variance

type	rational, p_1	rational, p_2	algebraic, p_1	algebraic, p_2
r_0a_1 (597)			$3.112\,10^{13}$	
r_0a_2 (639)			$2.785\,10^{13}$	$1.028\,10^{13}$
r_1a_0 (232)	$2.995\,10^{13}$			
r_1a_1 (770)	$2.903\,10^{13}$		$3.168\,10^{13}$	
r_1a_2 (743)	$2.914\,10^{13}$		$2.806\,10^{13}$	$1.069\,10^{13}$
r_2a_0 (88)	$3.157\,10^{13}$	$1.064\,10^{13}$		
r_2a_1 (231)	$2.853\,10^{13}$	$1.153\,10^{13}$	$2.984\,10^{13}$	
r_2a_2 (254)	$2.857\,10^{13}$	$1.024\,10^{13}$	$2.740\,10^{13}$	$1.078\,10^{13}$

Table 4.24: Required number of relations

type	rational, p_1	rational, p_2	algebraic, p_1	algebraic, p_2
r_0a_1 (597)			848	
r_0a_2 (639)			452	807
r_1a_0 (232)	865			
r_1a_1 (770)	892		884	
r_1a_2 (743)	872		418	812
r_2a_0 (88)	611	1027		
r_2a_1 (231)	481	807	975	
r_2a_2 (254)	481	832	427	788

The highest number of necessary relations is 1027 for r_2a_0 (for each type of relation we select the highest number); as this type had only 88 relations in the sieving test, this is not very accurate. If we look at r_0a_2, which has the same behavior of the large primes as r_2a_0, and occurred more frequently, we only need 807 relations. Therefore, if we allow a deviation of the average up to 10 % in 95 % of the cases, we should enlarge the sieving test in order to get at least 800 relations of r_2a_0. This agrees with the number 837 in Table 4.21, which was found after sieving 0.1 % of the data. This larger sieving test will give a better approximation of the averages and the behavior of the large primes. Consequently, we will get a better approximation of the total number of relations needed for factoring the given number.

4.6.2 Expected sieving area and sieving time

The same small sieving test we advised for estimating the size of the sample sieving test can be used for estimating the size of the sieving area. If the area is too small, we will not get enough relations and if it is too large, we will spend too much time on parts of the sieving area with a relative low yield. This affects especially line sieving; Figure 3.1 of [18] shows that the yield around the origin is the highest. We assume a rectangular sieving area with base A and height B, where the ratio A/B is prescribed by the choice of the polynomials. We increase (or decrease) both, to keep the suggested ratio of A/B. Once the area is estimated, we give an estimate of the total sieving time, based on the running time of the sieving test.

We concentrate on line sieving in this subsection. For lattice sieving a similar approach will work, if one selects the special primes in the sieving test carefully to represent the entire interval of special primes. Furthermore, we assume that we have already chosen the sieving bounds and that we only have to adjust the sieving area.

A simulation based on a small sieving test will give a rough approximation of the number of relations necessary for factoring the given number, but this estimate need not be close to the expected number of relations in the entire sieving area. We use the same small sieving test to decide the size of the next sieving test (cf. Subsection 4.6.1) and we get an estimate for the number of relations we will get on the entire

sieving area (just multiply the number of relations with the fraction *points in entire area / points in sieving test*). If this number of estimated relations in the sieving area agrees with the number of necessary relations, we only have to sieve more points in the next sieving test, else we should adjust the sieving area before starting a new sieving test.

To give an example of correcting the sieving area, we start for $19,183-$ with the following initial area: $A = [-7.5\text{M}, 7.5\text{M}]$ and $B = [1, 3\,011\,000]$. The polynomials and sieving parameters are the same as in Section 4.4. A sieving test used the lines with $b = i \times 30011$, $i = 1, 2, \ldots, 100$, which yielded 3540 relations. Based on these relations, we simulated relations and computed the oversquareness after removing singletons. A set of 18M relations gave an oversquareness of 95.40 %, and a set of 20M relations gave an oversquareness of 105.87 %. As we only get a rough impression of the relations (3540 relations represent only 0.0177 % of the possible 20M relations we should collect), we start with an estimate of 20M necessary relations.

The yield of the sieving test is 3540, which should be viewed as an unknown value in the interval $(3540 - \sqrt{3540}, 3540 + \sqrt{3540})$. This is based on the assumption that the yield of a sieving test follows a Poisson distribution and that we are within one standard deviation of the mean [34]. We expect that the yield of the entire area is 104.45M to 108.02M relations as we multiply the yield of the sieving test with 30011. Our first conclusion is that our area is much too large, as we expect to need about 20M relations.

Empirically we found that if we need a fraction α of the number of expected relations in the sieving area, we should multiply both A and B with (approximately) α. Therefore we multiply both A and B by $\frac{1}{5}$ in the new sieving test. If we want to work efficiently, we can simply take the relations from the first test that were found in the area $[-1.5\text{M}, 1.5\text{M}] \times [1, 600\,000]$. There are 704 relations fulfilling this requirement, thus we expect 20.33M to 21.92M relations in this area, which satisfies the first estimate of 20M relations. Based on only 704 relations, we repeated the simulation of 18M relations and 20M relations, which gave an oversquareness of 103.06 % and 113.13 %, respectively. The percentages changed, because the distribution of the large primes changed slightly, but the average in the model takes care of these changes.

We expect that our sieve area has the proper size, so we continue with a refinement of the sieving test. If we have bad luck, it will turn out that we have to change the area once more. To give an idea how good the prediction of 20M necessary relations is, we give in Table 4.25 of both sieving tests (of 3540 and 704 relations, named test 1 and test 2, respectively) the number of relations found and the needed number to get a good approximation of the average with $\varepsilon = 0.1\mu$ and $\text{P} = 0.05$, as explained in the previous subsection.

Table 4.25: Number of relations in sieving test and needed number

type	# in test 1	needed	# in test 2	needed
r_0a_1	539	599	112	539
r_0a_2	380	574	62	487
r_1a_0	375	584	86	594
r_1a_1	751	601	144	572
r_1a_2	494	651	79	654
r_2a_0	176	499	41	453
r_2a_1	326	551	71	581
r_2a_2	226	555	31	473

The two columns with the number of needed relations in the sieving test are quite close. As we need to refine the second test, we look for the highest quotient *needed number / number in test 2*. Here, we should sieve $473/31 \approx 15$ times as many points to get a good sieving test (test 3). We choose the prime 1999, which is closest to the quotient 30011/15 and we sieve the lines $b = i \times 1999$, $i = 1, 2, \ldots, 300$, over the complete width of 3M. This gives 9986 relations, so we expect 19.76M to 20.16M relations in the entire area. The analysis of the relations shows the following partition.

Table 4.26: Number of relations in sieving test and needed number

type	# in test 3	needed
r_0a_1	1486	539
r_0a_2	864	487
r_1a_0	1322	594
r_1a_1	2075	572
r_1a_2	1283	654
r_2a_0	570	453
r_2a_1	1005	581
r_2a_2	519	473

Compared with the number of needed relations, we see that we have enough relations of each type.

If we simulate 20M relations, we get an oversquareness of 109.19 %. In Subsection 4.4.1 we got an oversquareness of 99.22 % after simulating 20M relations, but there the sieving area was larger, which leads to larger polynomial values and probably a different distribution as the average values might change. To see if our new simulation gives the correct result, we took the original data of Subsection 4.4.1, extracted the relations located in the smaller area and removed singletons. There were 19.79M relations and they gave an oversquareness of 107.35 %. We simulated exactly the

same amount of relations and this gave an oversquareness of 108.17 %, which gives a relative difference of 0.76 %. So 20M relations are indeed sufficient to factor the number. Even 19M relations are enough, as 19M relations (based on the same test of 9986 relations) gave already an oversquareness of 103.61 %. Thus we can reduce the area even a bit further.

Finally we are able to give the expected sieving time. The sieving test found 9986 relations in 143.13 seconds. If we sieve the entire area ($[-1.5\text{M}, 1.5\text{M}] \times [1, 600\,000]$), we expect to find approximately 20M relations in $2000 \times 143.13 / 3600 = 79.52$ hours, when we sieve on the same machine. This is much less than the 154 hours in Table 4.9. There are two reasons for the difference: in Table 4.9 the area was 1.36 times larger, and the sieving time grows faster than linearly with the size of the sieving area and secondly we used many machines with different CPU speeds for the sieving.

4.6.3 Growth behavior of useful relations

In this subsection we look at the number of useful relations after singleton removal, as a function of the number of generated relations. For MPQS with three large primes, the growth is described as a kind of phase transition [30]. For NFS with at most two large primes on both the rational and algebraic side, the phenomenon of a sudden explosion (as defined in [17]) in the number of useful relations is described in [17]. In [19], the situation for hyperelliptic index calculus with two large primes is described, with the analysis of a simplified algorithm and the corresponding large primes graph. In this subsection, we investigate the interesting explosion phenomenon in NFS and demonstrate by some experiments that, by reducing the ratio of the large prime bound L and the factorbase bound F, the explosion may be avoided and the sieving time may be reduced.

We start with the following graph (Figure 4.13), which shows the growth behavior for $13,220+$, with horizontally the number of generated relations and vertically the number of relations after singleton removal (s.r.). It is hard to see, but the graph consists of two lines, one for the original data and one for the simulation.

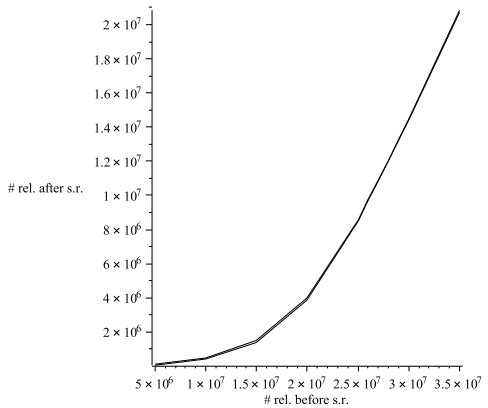

Figure 4.13: Growth behavior of useful relations

We notice the absence of an explosion or something that indicates a sudden change in the number of useful relations. Table 4.27, where we compare the data of $13,220+$ with the data in Table 3 of [17], gives the ratios of the different types of relations (with respect to the total number of relations used to factor the number). For completeness, $13,220+$ has 117 digits and the number in Table 3 of [17] has 116 digits.

Table 4.27: Ratios of the relations

type	[17], Table 3 (%)	13,220+(%)
r_0a_0	0.13	5.13
r_0a_1	1.46	15.43
r_0a_2	3.64	15.92
r_1a_0	1.02	6.68
r_1a_1	10.97	20.19
r_1a_2	27.26	20.79
r_2a_0	1.47	2.23
r_2a_1	15.81	6.72
r_2a_2	38.25	6.91

We see that the ratios are very different, as well as the used sieving bounds. For $13,220+$ we had $F = 30$M (1857860 primes in the factorbase) and $L = 400$M. Dodson and Lenstra used for the data in [17], Table 3 $\#FB_{\text{rat}} = 100\,001$, $\#FB_{\text{alg}} = 400\,001$, and $L = 2^{30} \approx 1074$M. Note that our factorbase bound is much larger and our large primes bound is 2.68 times smaller.

Motivated by this observation we performed sieving tests with different combinations of F and L, combined with the simulation of relations. We kept L constant at 400M and decreased F from 30M to 20M, 10M, 5M, and 3M. The growth behavior is indicated in Figure 4.14, where the line leftmost corresponds to F=30M and all lines are in decreasing order of the factorbase bound, so that the rightmost line corresponds with F=3M. We see the explosive behavior for F=3M and F=5M. The ratios show the same behavior as the data set in [17], so more than 80 % of the relations has three and four large primes in total. It is likely that it is much harder to combine these relations into cycles when there are not so many relations to choose from, as all three or four large primes have to match with primes in other relations. Once there are many relations, the matching becomes easier. If, on the other hand, a data set has more relations with only one or two large primes, the matching is easier and the growth of the useful relations is a more gradual growth. To see which behavior we should prefer in terms of shortest sieving time, we first show a graph with the oversquareness O_r (Figure 4.15). The lines (at height $O_r = 30$) are in the same order as in Figure 4.14.

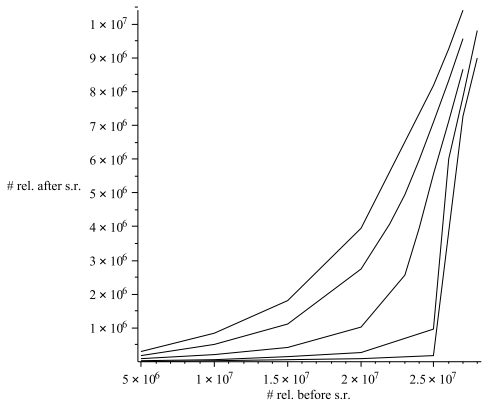

Figure 4.14: Growth behavior of the number of relations after singleton removal for $F =$ 30M, 20M, 10M, 5M, and 3M

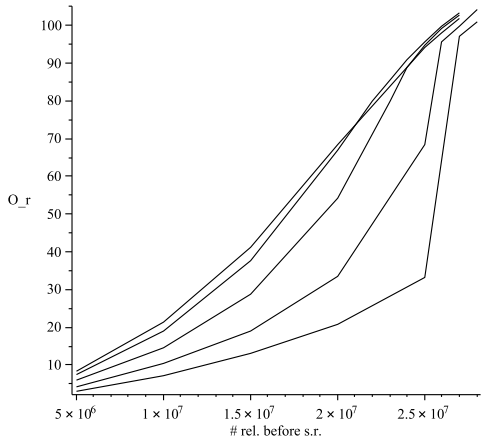

Figure 4.15: Growth behavior of the oversquareness for $F =$ 30M, 20M, 10M, 5M, and 3M

If we look at 100 % oversquareness, we see that as a function of the number of relations the line with $F =$ 20M reaches this point first, followed by $F =$ 10M, $F =$ 30M, $F =$ 5M, and $F =$ 3M. This does not automatically imply that $F =$ 20M gives the shortest sieving time. In Table 4.28 we give the number of relations needed to reach $O_r = 100$ and the expected sieving time, both based on a short sieving test.

Table 4.28: Expected sieving time

F	# rel. before s.r.	expected sieving time
3M	27.8M	312.8h
5M	27.0M	196.0h
10M	26.3M	126.5h
20M	26.1M	109.4h
30M	26.5M	107.2h

Although $F = 20$M needs the least amount of relations, it does not have the smallest sieving time. The most important influence on the shortest sieving time is the size of the sieving area, which was the smallest for $F = 30$M in this set of experiments. Another issue is the amount of cofactors (the composite factor of a polynomial value that is left after sieving) one needs to factor, as this takes some time as well.

Based on the data we show here and experiments on a few other data sets, we observe that the occurrence of an explosion in the number of useful relations depends largely on the factorbase bound and the large prime bound, as these bounds influence the ratios in which the different types of relations occur. As the two bounds move further apart, the probability of an explosive growth increases.

One of the reasons for still choosing a relatively small factorbase bound could be the physical limitations of the machine at hand. One of the authors of [17] told me that this was the case for their experiments and they tried to compensate it with L. With new record factorizations it is likely that the machine is the limiting factor again.

4.6.4 Oversquareness and matrix size

Here we study the relation between oversquareness of a set of relations and the resulting matrix size. More precisely, we want to know what is the optimal oversquareness. To get a good approximation of the size of the matrix, we perform the following two steps on subsets of the complete data set:

1. filter 1: remove all singletons, focus on primes larger than F.
2. filter 2: merge relations with mergelevel 9, focus on primes larger than F.

As explained in the thesis of Cavallar ([9], p. 54), mergelevel 9 indicates that at most 9-way merges are allowed to be executed. In the extreme case, if a prime ideal I occurs an odd number of times in 9 relations, we can choose 8 independent relation pairs out of the possible $\binom{9}{2}$ pairs. In each pair, the prime ideal I occurs an even number of times, hence can be part of a relation set to form a square. All details of the possible filter parameters and how they influence the filter algorithm can be found in Chapter 3 of [9]. Here we restrict our attention to the parameters which vary in

the two filter steps.

The relations left after the second filter determine the size of the matrix, as they represent the rows of the matrix. In order to get a fair comparison between the different subsets, we kept the parameters during the first and second filter fixed. Filter 1 has an extra parameter, **keep**. In the filter step we delete relations if the gap between the number of relations and the number of large primes after singleton removal is larger than the number **keep** which results in decreasing the oversquareness. The relations that are removed are part of so-called cliques, which can be seen as relations that stick together, i.e., if one of them is removed, the others will be removed as well as they become singletons. Using a suitable metric on the cliques, the heavy cliques are removed. In order to guarantee that the number of relations exceeds the number of large primes plus the number of primes in the factorbase, **keep** has to exceed the number of primes in the factorbase. In practice, **keep** is set at approximately $\frac{4}{3} \times (\pi(F_{\text{rat}}) + \pi(F_{\text{alg}}))$, in order to have more freedom in choosing relations in the next filter.

For the second step, there are two other important parameters. First, we set **maxpass** at 10. This fixes the number of shrinkage passes, during which all large primes are checked and possibly merged. Second, we set **maxdiscard** at approximately $\frac{1}{3} \times (\pi(F_{\text{rat}}) + \pi(F_{\text{alg}}))$, as this is the number of relations we can loose, without getting into trouble with covering primes in the factorbase, i.e., we aim at an oversquareness of 100 % after this filter.

For the relation between oversquareness and matrix size the *weight* of the matrix is relevant as well. The weight is defined as the number of 1's in the matrix. If the matrix is heavy, the linear algebra (we use block Lanczos) will need more memory and this may not fit on a given computer. Furthermore, the running time is linear in the weight of the matrix. The weight is influenced by the initial number of relations (how much freedom in choosing relations is available) and the filter strategies. It is difficult to optimize all the parameters to minimize the factorization time. Here, as a first approach, we look at the size of the resulting matrix as we vary the oversquareness.

To get a feeling for the relationship between oversquareness and matrix size, we start with the lattice sieve data set of $7,333-$ with $F = 16\,777\,215$ and $L = 250$M. The complete set consists of 25 112 543 relations and has $O_r = 136.64$. For filter 1 we set **keep** = 3 000 000. We give in Table 4.29 for different subsets the size of the subset, the oversquareness of the subset (after removing singletons, without the use of **keep**), the number of relations after filter 1, the number of relations after filter 2 and the number of large primes after filter 2, with between brackets the oversquareness after filter 2.

Table 4.29: Oversquareness and matrix size $(7,333-)$

subset	O_r	# rel. filter 1	# rel. filter 2	# l.p. filter 2 (O_r)
19M	109.52	9 424 147	4 015 131	1 148 335 (121.49)
21M	119.32	7 501 878	3 971 660	963 546 (127.30)
23M	128.13	6 721 073	3 860 995	851 142 (128.38)
25M	136.20	6 294 741	3 765 882	758 009 (129.21)

As there is still a difference of about 3M between the number of relations and the number of large primes (which is reflected in the large numbers of oversquareness given in the last column), we add an extra parameter to filter 2. We set **maxrels** at 4.0 instead of the standard 10.0, so a relation set has a maximum weight of 4.0. Relation sets that are too heavy, will be removed. This value of **maxrels** is much lower than what we normally choose, but now we only look at the primes above F. In Table 4.30, we give the size of the subset, the number of shrinkage passes (how often the file is read after removing part of the relations and start again with combining relations), the number of relations after filter 2 and the number of large primes after filter 2.

Table 4.30: Oversquareness and matrix size $(7,333-)$

subset	# passes	# rel. filter 2	# l.p. filter 2 (O_r)
19M	8	3 538 379	1 290 950 (102.64)
21M	10	2 969 426	748 917 (102.21)
23M	10	2 773 822	646 840 (101.08)
25M	10	3 180 923	669 176 (112.57)

The last entry in Table 4.30 is higher than expected, but this is due to the fact that **maxdiscard** is not reached after 10 passes, i.e. we can still remove relations and keep enough excess to cover the primes in the factorbase.

If we look at the number of relations after filter 1 (in Table 4.29), we see a non-linear decrease, where we gain the most when we increase the initial subset from 19M to 21M relations. The same effect is visible in Table 4.30, as the number of relations after filter 2 decreases from 3 538 379 relations to 2 969 426 relations when the size of the initial subset increases with two million relations to 21M relations.

To see if this behavior occurs more often, we look at another data set. We take the line sieving data set of $13,220+$ with $F = 30$M and $L = 2^{29} \approx 537$M. The complete data set has 48 387 564 relations and an oversquareness of 132.80 %. We set **keep** at 5 000 000, and **maxrels** at 4.0. In Table 4.31 we give the size of the subset, the oversquareness of this subset after removing singletons, the number of relations after filter 1, the number of relations after filter 2 with between brackets the number of

passes if this differs from 10, and the number of large primes after filter 2, with the oversquareness after filter 2 between brackets.

Table 4.31: Oversquareness and matrix size (13,220+)

subset	O_r	# rel. filter 1	# rel. filter 2	# l.p. filter 2 (O_r)
34M	106.29	12 226 490	5 024 040	1 303 589 (100.39)
36M	111.29	12 894 358	4 122 961 (7)	403 999 (100.44)
38M	115.83	11 056 646	3 784 399	56 501 (100.72)
40M	120.04	10 076 131	4 064 445	58 911 (108.10)
42M	124.00	9 410 076	4 387 683	68 784 (116.39)
44M	127.78	8 919 825	4 591 032	64 559 (121.92)
46M	131.41	8 478 465	4 703 480	65 659 (124.87)
48M	134.94	8 177 874	4 865 822	59 813 (129.38)

For the subset of 34M relations, we had to adjust **maxdiscard** in filter 2, as the difference between the number of relations and the number of large primes after filter 1 was only 4.4M instead of the required 5M (by **keep**).

In the column of the number of relations after filter 1 (Table 4.31), we see the highest decrease when the subset increases from 36M to 38M relations. However, if we look at the increase in oversquareness after filter 2, we see the highest increase from 100.72 % to 108.10 % to 116.39 %, which coincides with a subset of about 40M relations. Since the total sieving time increases faster than linearly, it is not useful to go beyond 40 M relations.

Note that the subset of 40M relations has an oversquareness of 120.04 % after removing only singletons, whereas the subset of 21M relations of $7,333-$ (which seems to be optimal for this number in terms of matrix size vs. number of relations) has an oversquareness of 119.32 % after removing singletons. It may well be that an oversquareness of about 120 % after removing singletons is a good estimate for a relatively small matrix. As we ignored the weight of the matrix, this is only a rough indication of what a good oversquareness after removing singletons is for optimizing the running time of the sieving step and the linear algebra step together.

Chapter 5

Conclusions and Suggestions for Future Research

Throughout this thesis we have worked on different ways to obtain a good estimate of the sieving time in advance. We started with a theoretical approach in Chapter 2 and compared it with practical results in Chapter 3. This showed that there was a discrepancy between theoretical and practical results. In Chapter 4 we gave a practical approach for estimating the sieving time of NFS and the resulting estimates for the sieving time are good. In this chapter we summarize the main results and give conclusions, as well as suggestions for further research.

5.1 Smooth and semismooth numbers

In Chapter 2 we focused on what is theoretically known about the expected number of smooth and semismooth numbers. Our main contribution is stated in Theorem 7 (Subsection 2.4.2): it gives the main term, the second order term and error term of the asymptotic expansion in powers of $1/\log x$, for $x \to \infty$, of the expected number of k-semismooth numbers $\leq x$ ($k = 1, 2, \ldots$) with possibly different upper bounds on the large primes where the main and second order term are given explicitly as functions of the Dickman ρ function.

In Chapter 3 we compared our theoretical results with practical data, obtained during the sieving step of the factoring algorithms MPQS and NFS. One of the conclusions is that for numbers of approximately 100 decimal digits, the second order term adds about 10 % to the main term. In general, this is a considerable contribution that should be taken into account. However, in our experiments we see sometimes relatively big differences between theory and practice. Possible causes might depend on the use of asymptotic formulas, the implementation of MPQS and NFS at hand, and a too crude estimate of the polynomial values. A second conclusion is that it may be beneficial to use different upper bounds on the two large primes of 2-semismooth num-

bers in MPQS. We compared the timings by keeping the product L_1L_2 constant and only allow 2-semismooth relations with one prime $\leq L_1$ and the other prime $\leq L_2$. An improvement of our approach might be to keep as well 2-semismooth numbers with both large primes between L_1 and L_2 as long as their product $\leq L_1L_2$. These additional relations will influence the optimal values of L_1 and L_2, which should be investigated further. In order to be able to choose optimal bounds, we need to know how to predict the sieving time of MPQS. This is due to the fact that different bounds lead to different frequencies of (semi)smooth numbers. The rate at which the relations are found differs as well per type of relation, thus we need to find out in advance how many relations are necessary to factor the number at hand. As far as we know, there is no good way of predicting the number of necessary relations [28] before the sieving is started, hence no good estimate of the sieving time. It might be possible to adjust the developed method, i.e., simulating the large primes, for the Number Field Sieve. The efficacy of this treatment is left for further research.

As some factorers already have started to use 3-semismooth numbers, it would be useful to extend our experiments from 1- and 2-semismooth numbers to 3- and 4-semismooth numbers: we expect the generalization to be straightforward, but the implementation may be cumbersome. Another object of study is the optimization of the integration regions for computing the expected number of relations, as we noticed in Section 3.4 a big improvement by dividing the sieving region into only eight subregions.

5.2 Simulating the sieving

In Chapter 4 we presented a method for predicting the number of relations needed for factoring a given number with NFS, and, subsequently, the required sieving time. This method is based on a short sieving test; from the relations found during this test we obtain a good model of the distribution of the large primes in the relations. Next we cheaply simulate all the relations needed for the factorization, as we use easily computable numbers with the same distribution as prescribed by our model.

Our experiments suggest us to use two different models to simulate data sets, Case I and Case II. If the large prime bound is at most 20 times the factorbase bound, Case I seems to be preferred. If the large prime bound is at least 100 times the factorbase bound, Case II seems to be the better choice. As extra support for making the proper choice, graphs of the large primes of the different types of relations can be very helpful, see for instance Figures 4.1, 4.2, 4.3 and 4.4. The choice of the correct model seems to be independent of the implementation of line sieving or lattice sieving.

The simulation of the relations combined with singleton removal shows that our estimation of the oversquareness is within 2 % of the real data in Case I; in Case II it is approximately 5 %. By our simulation we cheaply obtain a good estimate for the number of necessary relations for factoring a given number on a given computer, and hence of the actual computing time. Therefore, this method is a useful tool for optimizing parameters in the number field sieve.

Based on a small sieving test, we choose both the size of the next (proper) sieving test and the size of the sieving area, as a first simulation gives an indication of the necessary number of relations for factoring the given number. If the sieve area needs a big adjustment, a second small sieving test on this new area is helpful to check correctness, before starting with the proper sieving test.

We compared different combinations of factorbase bounds and large prime bounds with the number of relations after singleton removal. This showed that different types of growth behavior of the number of relations after singleton removal are possible, but for line sieving a gradual growth seemed to lead to less sieving time than an explosive growth. Therefore the factorbase bound should not be chosen too small.

We also looked at the influence of the oversquareness (ratio of the number of relations to the number of large primes, cf. Section 4.3) on the size of the resulting matrix. There are many different filter strategies to get from the initial data set to a smaller subset, which makes it difficult to perform a good analysis. We found that, as a starting point, an oversquareness of about 120 % gives a much smaller matrix than 100 % oversquareness. A larger percentage oversquareness will decrease the matrix further, but the decrease becomes less and it takes considerably more time to generate the extra relations.

It is desirable to run more experiments with both line sieving and lattice sieving, while varying the size of N, F, and L, in order to see if more models are required for simulating the various data sets and which model is the best in a given situation. Once this is known, we can use ideas from statistics to choose the correct model, given the relations of a sieving test. With these extra data sets, we will be able to get a better understanding of the relation between oversquareness and matrix size. This will get us closer to the goal of developing a tool to automatically determine bounds F and L that optimize the overall effort for relation collection and matrix processing with respect to the available resources.

As mentioned in Section 5.1 the experiments should include cases with three large primes as well. Especially with the present record factorization attempt (RSA768), relations with 3 large primes on each side are collected. This data set will provide a good test case for these new models, whose development is left for future research.

Another subject for further research is the matching behavior of the large primes. We found a reason for an explosive growth behavior (cf. Subsection 4.6.3), yet we do not know exactly how the number of relations after singleton removal grows and this remains subject of further research.

Bibliography

[1] Aggarwal, D., Maurer, U.: Factoring is equivalent to generic RSA. Cryptology ePrint Archive, Report 2008/260 (2008)

[2] Alford, W.R., Pomerance, C.: Implementing the self initializing quadratic sieve on a distributed network. Number Theoretic and Algebraic Methods in Computer Science, Proc. Internat. Moscow Conf. 1993, World Scientific 163–174

[3] K. Aoki, J. Franke, T. Kleinjung, A.K. Lenstra, D.A. Osvik: A kilobit special number field sieve factorization. Adv. Crypt. - ASIACRYPT 2007, LNCS 4833 (2007) 1–12

[4] Bach, E., Peralta, R.: Asymptotic semismoothness probabilities. Math. Comp. **65** (1996) 1701–1715

[5] Bernstein, D.J.: Arbitrarily tight bounds on the distribution of smooth integers. http://cr.yp.to/papers/psi.pdf (2001)

[6] Boender, H.: Factoring Large Integers with the Quadratic Sieve. Ph.D. thesis, University of Leiden (1997)

[7] Breiman, L.: Statistics: With a View Toward Applications. Houghton Mifflin Company Boston (1973)

[8] de Bruijn, N.G.: On the number of positive integers $\leq x$ and free of prime factors $> y$. Proc. Kon. Ned. Akad. Wet. **A54** = Indag. Math. **13** (1951) 50–60

[9] Cavallar, S.H.: On the Number Field Sieve Integer Factorisation Algorithm. Ph.D. thesis, University of Leiden (2002)

[10] Contini, S.P.: Factoring integers with the Self-Initializing Quadratic Sieve. Ph.D. thesis, University of Georgia (1997)

[11] Cooley, J.W., Tukey, J.W.: An algorithm for the machine calculation of complex Fourier series. Math. Comp. **19** (1965) 297–301

[12] Coppersmith, D.: Solving linear equations over GF(2): block Lanczos algorithm. Linear Algebra Appl. **192** (1993) 33–60

[13] Coppersmith, D.: Solving homogeneous linear equations over GF(2) via block Wiedemann algorithm. Math. Comp. **62** (1994) 333–350

[14] Crandall, R.E., Pomerance, C.: Prime Numbers: a Computational Perspective. Springer-Verlag, New York (2001)

[15] Davis, J.A., Holdridge, D.B.: Factorization using the quadratic sieve algorithm. Tech. Report SAND 8301346, Sandia National Laboratories, Albuquerque, NM (1983)

[16] Dekking, F.M., Kraaikamp, C., Lopuhaä, H.P., Meester, L.E.: A Modern Introduction to Probability and Statistics: Understanding Why and How. Springer Texts in Statistics. Springer-Verlag London, Ltd., London (2005)

[17] Dodson, B., Lenstra, A.K.: NFS with four large primes: an explosive experiment. Advances in cryptology—CRYPTO '95 (Santa Barbara, CA, 1995), Lecture Notes in Comput. Sci. **963**, Springer, Berlin (1995) 372–385

[18] Elkenbracht-Huizing, M.: Factoring Integers with the Number Field Sieve. Ph.D. thesis, University of Leiden (1997)

[19] Gaudry, P., Thomé, E., Thériault, N., Diem, C.: A double large prime variation for small genus hyperelliptic index calculus. Math. Comp. **76** (2007) 457–492

[20] Hardy, G.H., Wright, E.M.: An Introduction to the Theory of Numbers. Oxford University Press, New York (1979)

[21] Hildebrand, A.: On the number of positive integers $\leq x$ and free of prime factors $> y$. J. Number Theory **22** (1986) 289–307

[22] Hildebrand, A., Tenenbaum, G.: Integers without large prime factors. J. Théor. Nombres Bordeaux **5(2)** (1993) 411–484

[23] Kleinjung, Th.: On polynomial selection for the general number field sieve. Math. Comp. **75** (2006) 2037–2047

[24] Knuth, D.E., Trabb Pardo, L.: Analysis of a simple factorization algorithm. Theoret. Comput. Sci. **3(3)** (1976/77) 321–348

[25] Lambert, R.: Computational Aspects of Discrete Logarithms. Ph.D. thesis, University of Waterloo (1996)

[26] Lenstra, A.K.: QS, software package (LIP based quadratic sieve). (1995)

[27] Lenstra, A.K., Lenstra, H.W., Jr. (Eds.): The Development of the Number Field Sieve. Lecture Notes in Math., vol. 1554, Springer, Berlin (1993)

[28] Lenstra, A.K., Manasse, M.S.: Factoring with two large primes. Math. Comp. **63** (1994) 785–798

[29] Lenstra, A.K., Tromer, E., Shamir, A., Dodson, B., Hughes, J., Leyland, P.: Factoring estimates for a 1024-bit RSA modulus. Advances in cryptology—ASIACRYPT 2003, Lecture Notes in Comput. Sci. **2894**, Springer, Berlin (2003) 55–74

[30] Leyland, P., Lenstra, A.K., Dodson, B., Muffett, A., Wagstaff, Jr., S.S.: MPQS with three large primes. Algorithmic number theory (Sydney, 2002), Lecture Notes in Comput. Sci. **2369**, Springer, Berlin (2002) 446–460

[31] Maple, Waterloo Maple Inc., Waterloo, Canada

[32] Montgomery, P.L.: A survey of modern integer factorization algorithms. CWI Quarterly, **7/4** (1994) 337–366

[33] Montgomery, P.L.: A block Lanczos algorithm for finding dependencies over GF(2). Advances in cryptology—EUROCRYPT '95 (Saint-Malo, 1995), Lecture Notes in Comput. Sci. **921**, Springer, Berlin (1995) 106–120

[34] P.L. Montgomery, private communication

[35] Murphy, B.: Polynomial Selection for the Number Field Sieve Integer Factorisation Algorithm. Ph.D. thesis, Australian National University (1999)

[36] Norton, K.K.: Numbers with Small Prime Factors, and the Least kth Power Non-Residue. Memoirs of the American Mathematical Society, No. 106, American Mathematical Society, Providence, R.I. (1971)

[37] Panaitopel, L.: A formula for $\pi(x)$ applied to a result of Koninck-Ivić. Nieuw Arch. Wiskunde, **5/1** (2000) 55–56

[38] Parsell, S.T., Sorensen, J.P. Fast bounds on the distribution of smooth numbers. Algorithmic number theory (Berlin 2006), Lecture Notes in Comput. Sci., 4076, Springer, Berlin (2006) 168–181

[39] Pomerance, C.: The quadratic sieve factoring algorithm. Advances in cryptology (Paris, 1984), Lecture Notes in Comput. Sci. **209**, Springer, Berlin (1985) 169–182

[40] Ramaswami, V.: On the number of positive integers less than x and free of prime divisors greater than x^c. Bull. Amer. Math. Soc. **55** (1949) 1122–1127

[41] Ramaswami, V.: The number of positive integers $\leq x$ and free of prime divisors $> x^c$, and a problem of S.S. Pillai. Duke Math. J. **16** (1949) 99–109

[42] Rankin, R.A.: The difference between consecutive prime numbers. J. London Math. Soc. **13** (1938) 242–247

[43] Riesel, H.: Prime Numbers and Computer Methods for Factorization. Progress in Mathematics, vol. 57, Birkhäuser Boston, Inc., Boston, MA (1985)

[44] Rivest, R.L., Shamir, A., Adleman, L.: A method for obtaining digital signatures and public-key cryptosystems. Comm. ACM **21(2)** (1978) 120–126

[45] Silverman, R.D.: The multiple polynomial quadratic sieve. Math. Comp. **48** (1987) 329–339

[46] Suzuki, K.: Approximating the number of integers without large prime factors. Math. Comp. **75** (2005) 1015–1024

[47] Tenenbaum, G.: A rate estimate in Billingsley's theorem for the size distribution of large prime factors. Quart. J. Math. **51** (2000) 385–403

[48] Vinogradov, J.M.: On the bound of the least non-residue of nth powers. Trans. Amer. Math. Soc. **29** (1927) 218–226

[49] Wiedemann, D.H.: Solving sparse linear equations over finite fields. IEEE Trans. Inform. Theory **32(1)** (1986) 54–62

[50] Zhang, C.: An extension of the Dickman function and its application Ph.D. thesis, Purdue University (2002)

Samenvatting

Het ontbinden van een getal in priemfactoren is een eeuwenoud probleem. Het is mogelijk om van twee gegeven grote getallen snel het product te berekenen, maar in het algemeen is het lastig om uit een gegeven product van twee grote priemgetallen de beide priemfactoren te bepalen. Vanwege deze moeilijkheid worden zulke grote getallen (meestal het product van twee priemgetallen van dezelfde orde van grootte) gebruikt in het RSA encryptiealgoritme om berichten te versleutelen en veilig te versturen. Veilig wil zeggen: zonder dat de berichten begrepen of ongemerkt veranderd kunnen worden door derden. Om te weten welke grootte nog veilig is, wordt onderzoek gedaan naar algoritmes die grote getallen ontbinden in priemfactoren.

In dit proefschrift worden twee zulke algoritmes bestudeerd en wel MPQS (Multiple Polynomial Quadratic Sieve) en NFS (Number Field Sieve). In beide algoritmes worden eerst polynomen gekozen. Vervolgens worden polynoomwaarden ontbonden in priemfactoren in zoverre ze samengesteld zijn uit priemgetallen kleiner dan een voorgeschreven grens F. Zulke waarden noemen we gladde getallen. Aanvullend worden ook bijna-gladde getallen gebruikt. Dit zijn getallen die alle priemfactoren beneden F hebben met uitzondering van één of twee priemfactoren tussen F en een hogere grens L. De priemgetallen $\leq F$ noemen we kleine priemgetallen, die tussen F en L grote priemgetallen. Deze (bijna-)gladde polynoomwaarden leiden tot zogenaamde relaties die bestaan uit paren getallen $(x, f(x))$ bij MPQS en uit paren $(f_1(x, y), f_2(x, y))$ bij NFS. Dit proces van het verzamelen van geschikte relaties noemen we zeven en het is de meest tijdrovende stap van beide algoritmes. Het aantal grote priemfactoren in bijna-gladde getallen kan worden uitgebreid tot k priemfactoren tussen F en L, waarbij k een willekeurig natuurlijk getal is. Een andere generalisering is het gebruik van verschillende bovengrenzen voor de verschillende grote priemfactoren in plaats van een vaste bovengrens L.

Aangezien er veel relaties nodig zijn om het getal te kunnen ontbinden, is het een natuurlijke vraag om uit te zoeken hoeveel getallen onder een gegeven grens (bijna-) glad zijn. In hoofdstuk 2 geven we een theoretische analyse van het te verwachten aantal (bijna-)gladde getallen. Dit is een uitbreiding van resultaten van, onder anderen, De Bruijn, Ramaswami, Bach en Peralta, en Lambert. De benaderingen worden uitgedrukt in een hoofdterm en een restterm of in een hoofdterm, een tweede orde term en een restterm. In hoofdstuk 3 vergelijken we de theoretische aantallen met de praktische aantallen, zoals we die tegenkomen bij implementaties van MPQS en NFS.

We gaan na hoe groot de invloed van de tweede orde term is en hoe groot het voordeel van het kiezen van verschillende bovengrenzen. Uit de analyses blijkt dat het gebruik van de theoretisch berekende aantallen nog niet tot een bruikbare schatting van de benodigde zeeftijd leidt.

Om toch een goede indicatie van de zeeftijd te krijgen, hebben we voor NFS een nieuwe methode ontwikkeld om de zeeftijd te schatten. Deze methode is gebaseerd op een korte zeeftest, waarna de dichtheid van de grote priemgetallen in de hiermee verkregen relaties wordt bepaald. In hoofdstuk 4 wordt een model gepresenteerd waarmee we in zeer korte tijd veel relaties simuleren, doordat we, in plaats van met grote priemgetallen, rekenen met getallen die sneller berekenbaar zijn en dezelfde dichtheid hebben als de grote priemgetallen uit de zeeftest. Met de zo gevonden gesimuleerde relaties bepalen we hoeveel relaties nodig zijn om het getal te ontbinden en met experimenten laten we zien dat hiermee het noodzakelijke aantal relaties tot op 2 % nauwkeurig voorspeld kan worden. Daardoor wordt het mogelijk om de parameters van NFS te optimaliseren. Omdat het aantal relaties rechtstreeks gekoppeld is aan de tijd die nodig is voor het zeven, is het mogelijk om na een korte zeeftest de voor factorisatie benodigde rekentijd nauwkeurig te schatten en zo laag mogelijk te houden.

We geven ook een tweede model in hoofdstuk 4. Het verschil met het eerste model is de distributie van de grote priemgetallen in de relaties. Deze distributie hangt samen met de gebruikte grenzen F en L. De keuze voor het meest geschikte model is dan ook afhankelijk van F en L.

Een ander aspect is het bepalen van de grootte van de zeeftest. Als de test te groot is, kost het te veel tijd, maar als de test te klein is, is de voorspelling niet nauwkeurig. We beginnen dan ook met een kleine zeeftest (0.01 % van het zeefgebied) en door Chebyshev's ongelijkheid toe te passen op de data uit de zeeftest krijgen we informatie over hoe groot de zeeftest zou moeten zijn. Ook al is de eerste zeeftest erg klein, we krijgen met behulp van een simulatie al een eerste schatting van het aantal relaties dat nodig is. Hiermee kunnen we nagaan of het zeefgebied voldoende groot is. Met een nieuwe zeeftest van de juiste grootte en een zeefgebied van de goede omvang krijgen we een goede voorspelling van het aantal relaties dat nodig is om het getal te factoriseren. En indien nodig wordt het zeefgebied opnieuw aangepast.

Verder kijken we nog naar de toename van het aantal nuttige relaties (niet alle gevonden relaties zijn bruikbaar) ten opzichte van het aantal gevonden relaties en naar de omvang van de matrix die in een latere stap van NFS gemaakt wordt. De omvang hangt samen met het aantal relaties en een toename van het aantal relaties zorgt voor een kleinere matrix, maar het precieze verband is niet bekend.

In hoofdstuk 5 geven we conclusies en suggesties voor verder onderzoek.

Acknowledgements

I thank Arjen Lenstra for suggesting the idea to predict the sieving effort by simulating relations on the basis of a short sieving test and Rob Tijdeman for inspiring discussions and eye for detail: even the slightest hand waving would be detected. Furthermore, I thank Herman te Riele for making me familiar with big computations, such as executing the CWI Number Field Sieve on many machines.

I thank Marie-Colette van Lieshout for suggesting several statistical models for simulating the large primes, including the model which is used in Case I for $r_1 a_0$ and Dag Arne Osvik for providing the singleton removal code for relations written in a special format. I further thank Bruce Dodson and Thorsten Kleinjung for sharing data sets, and Thorsten for making available his lattice siever code as well.

I am grateful to all my colleagues, family, friends and neighbours for beautiful and inspiring moments and their support during the years of my Ph.D. research. Or as we say in Den Ham: "Bedankt veur alns en goodgoan!"

Curriculum Vitae

Willemien Ekkelkamp werd geboren op 9 augustus 1978 te Almelo. In 1996 behaalde zij haar VWO-diploma aan het Noordik te Almelo. Daarna heeft ze de opleiding leraar voortgezet onderwijs 2^e graads wiskunde gedaan aan de Christelijke Hogeschool Windesheim te Zwolle. Als afgestudeerd docente wiskunde heeft zij in het schooljaar 2000-2001 les gegeven aan de Christelijke Scholengemeenschap Reggesteyn te Nijverdal en Rijssen. Na dit jaar is ze wiskunde gaan studeren aan de Rijksuniversiteit Groningen, waar ze in 2004 afstudeerde. Ze schreef haar afstudeerscriptie "On the computation of Lie symmetries" onder begeleiding van prof. dr. M. van der Put.

In februari 2004 begon zij als Onderzoeker in Opleiding bij het Centrum voor Wiskunde en Informatica te Amsterdam en bij de Universiteit Leiden. Haar onderzoek werd gefinancierd door de Nederlandse Organisatie voor Wetenschappelijk Onderzoek. Tijdens dit onderzoek werd zij begeleid door dr. ir. H.J.J. te Riele, prof. dr. R. Tijdeman en prof. dr. A.K. Lenstra.

Vanaf 13 oktober 2008 is Willemien werkzaam als system analysis engineer bij Thales NL te Hengelo.