Algorithmic Number Theory – Proceedings of ANTS-III
(June 21–25, 1998, Portland, Oregon, USA)
J. Buhler (Ed.), vol. 1423 of Lecture Notes in Computer Science, pages 151–168

A Montgomery-like Square Root for the Number Field Sieve

Phong Nguyen

Phong.Nguyen@ens.fr
http://www.dmi.ens.fr/~pnguyen/
Ecole Normale Supérieure
Laboratoire d'Informatique
45, rue d'Ulm
F - 75230 Paris Cedex 05

Abstract. The Number Field Sieve (NFS) is the asymptotically fastest factoring algorithm known. It had spectacular successes in factoring numbers of a special form. Then the method was adapted for general numbers, and recently applied to the RSA-130 number [6], setting a new world record in factorization. The NFS has undergone several modifications since its appearance. One of these modifications concerns the last stage: the computation of the square root of a huge algebraic number given as a product of hundreds of thousands of small ones. This problem was not satisfactorily solved until the appearance of an algorithm by Peter Montgomery. Unfortunately, Montgomery only published a preliminary version of his algorithm [15], while a description of his own implementation can be found in [7]. In this paper, we present a variant of the algorithm, compare it with the original algorithm, and discuss its complexity.

1 Introduction

The number field sieve [8] is the most powerful known factoring method. It was first introduced in 1988 by John Pollard [17] to factor numbers of form $x^3 + k$. Then it was modified to handle numbers of the form $r^e - s$ for small positive r and $|s|$: this was successfully applied to the Fermat number $F_9 = 2^{512} + 1$ (see [11]). This version of the algorithm is now called the special number field sieve (SNFS) [10], in contrast with the general number field sieve (GNFS) [3] which can handle arbitrary integers. GNFS factors integers n in heuristic time $\exp\Big((c_g + o(1)) \ln^{1/3} n \ln^{2/3} \ln n\Big)$ with $c_g = (64/9)^{1/3} \approx 1.9$.

Let n be the composite integer we wish to factor. We assume that n is not a prime power. Let $\mathbb{Z}_n$ denote the ring $\mathbb{Z}/n\mathbb{Z}$. Like many factoring algorithms, the number field sieve attempts to find pairs $(x, y) \in \mathbb{Z}_n^2$ such that $x^2 \equiv y^2 \pmod n$. For such a pair, $\gcd(x - y, n)$ is a nontrivial factor of n with a probability of at least $\frac{1}{2}$. The NFS first selects a primitive polynomial $f(X) = \sum_{j=0}^{d} c_j X^j \in \mathbb{Z}[X]$ irreducible over $\mathbb{Z}$, and an integer m with $f(m) \equiv 0 \pmod n$. Denote by $F(X, Y) = Y^d f(X/Y)$ in $\mathbb{Z}[X, Y]$ the homogenous form of f. Let $\alpha \in \mathbb{C}$ be a root of f, and $\mathbb{K} = \mathbb{Q}(\alpha)$ be the corresponding number field. There is a natural ring homomorphism ϕ from $\mathbb{Z}[\alpha]$ to $\mathbb{Z}_n$ induced by $\phi(\alpha) \equiv m \pmod n$. We will do as if ϕ mapped the whole $\mathbb{K}$. If ever $\phi(\beta)$ is not defined for some $\beta \in \mathbb{K}$, then we have found an integer not invertible in $\mathbb{Z}_n$, and thus, a factor N of n which should not be trivial. If $n' = n/N$ is prime, the factorization is over, and if not, we replace n by n', and ϕ by ϕ' induced by $\phi'(\alpha) \equiv m \pmod{n'}$.

By means of sieving, the NFS finds several integer pairs (a_i, b_i) and a finite nonempty set S such that $\prod_{i\in S}(a_i - b_i\alpha)$ and $\prod_{i\in S}(a_i - b_i m)$ are squares in $\mathbb{K}$ and in $\mathbb{Z}$, respectively. We have $\phi\left(\prod_{i\in S}(a_i - b_i\alpha)\right) \equiv \prod_{i\in S}(a_i - b_i m) \pmod n$, therefore

$$\left[\phi\left(\sqrt{\prod_{i\in S}(a_i - b_i\alpha)}\right)\right]^2 \equiv \left[\sqrt{\prod_{i\in S}(a_i - b_i m)}\right]^2 \pmod n$$

after extracting the square roots, which gives rise to a suitable pair (x, y). The NFS does not specify how to evaluate these square roots. The square root of the integer $\prod_{i\in S}(a_i - b_i m) \mod n$ can be found using the known prime factorizations of each $a_i - b_i m$. But extracting the square root of $\prod_{i\in S}(a_i - b_i\alpha)$ is much more complicated and is the subject of this paper. We note $\gamma = \prod_{i\in S}(a_i - b_i\alpha)$. The following facts should be stressed:

- the cardinality $|S|$ is large, roughly equal to the square root of the run time of the number field sieve. It is over 10^6 for n larger than 100 digits.
- the integers a_i, b_i are coprime, and fit in a computer word.
- the prime factorization of each $F(a_i, b_i)$ is known.
- for every prime number p dividing c_d or some $F(a_i, b_i)$, we know the set $R(p)$ consisting of roots of f modulo p, together with ∞ if p divides c_d.

The remainder of the paper is organized as follows. In Section 2, we review former methods to solve the square root problem, one of these is used in the last stage of the algorithm. Section 3 presents a few definitions and results. In Section 4, we describe the square root algorithm, which is a variant of Montgomery's original algorithm, and point out their differences and similarities. We discuss its complexity in Section 5. Finally, we make some remarks about the implementation in Section 6, and the appendix includes the missing proofs.

2 Former methods

UFD method. If α is an algebraic integer and the ring $\mathbb{Z}[\alpha]$ is a unique factorization domain (UFD), then each $a_i - b_i\alpha$ can be factored into primes and units, and so can be γ, which allows us to extract a square root of γ. Unfortunately, the ring $\mathbb{Z}[\alpha]$ is not necessarily a UFD for the arbitrary number fields GNFS encounters. And even though $\mathbb{Z}[\alpha]$ is a UFD, computing a system of fundamental units is not an obvious task (see [4]). The method was nevertheless applied with success to the factorization of F_9 [11].

Brute-force method. One factorizes the polynomial $P(X) = X^2 - \gamma$ over $\mathbb{K}[X]$. To do so, one has to explicitly write the algebraic number γ, for instance by expanding the product: one thus gets the (rational) coefficients of γ as a polynomial of degree at most $d-1$ in α. But there are two serious obstructions: the coefficients that one keeps track of during the development of the product have $\mathcal{O}(|S|)$ digits. Hence, the single computation of the coefficients of γ can dominate the cost of the whole NFS. And even if we are able to compute γ, it remains to factorize $P(X)$.

One can overcome the first obstruction by working with integers instead of rationals: let $\hat{f}(X)$ be the monic polynomial $F(X, c_d)$, and $\hat{\alpha}$ be the algebraic integer $c_d\alpha$ which is a root of $\hat{f}$. If γ is a square in $\mathbb{K}$ then $\gamma' = c_d^{2\lceil |S|/2 \rceil} \hat{f}'(\hat{\alpha})^2\gamma$ is a square in $\mathbb{Z}[\hat{\alpha}]$, where $\hat{f}'$ denotes the formal derivative of $\hat{f}$. It has integral coefficients as a polynomial of degree at most $d-1$ in $\hat{\alpha}$, and these can be obtained with the Chinese Remainder Theorem, using several inert primes (that is, f is irreducible modulo this prime) if there exist inert primes (which is generally true). This avoids computations with very large numbers. However, one still has to factorize the polynomial $Q(X) = X^2 - \gamma'$, whose coefficients remain huge, so the second obstruction holds. Furthermore, a large number of primes is required for the Chinese Remainder Theorem, due to the size of the coefficients.

Couveignes's method. This method overcomes the second obstruction. If f has odd degree d, Couveignes [5] remarks that one is able to distinguish the two square roots of any square in $\mathbb{K}$, by specifying its norm. Let $\sqrt{\gamma'}$ be the square root with positive norm. Since the prime factorization of $\mathcal{N}(\gamma')$ is known, the integer $\mathcal{N}(\sqrt{\gamma'})$ can be efficiently computed modulo any prime q. If q is inert then $\sqrt{\gamma'} \pmod q$ can be computed after expanding $\gamma' \pmod q$. From the Chinese Remainder Theorem, one recovers the coefficients of $\sqrt{\gamma'} \in \mathbb{Z}[\hat{\alpha}]$. One can show that the complexity of the algorithm is at best $\mathcal{O}(M(|S|) \ln |S|)$, where $M(|S|)$ is the time required to multiply two $|S|$-bit integers. The algorithm appears to be impractical for the sets S now in use, and it requires an odd degree.

Montgomery's strategy [15, 14, 7] can be viewed as a mix of UFD and brute-force methods. It bears some resemblance to the square root algorithm sketched in [3] (pages 75-76). It works for all values of d, and does not make any particular assumption (apart from the existence of inert primes) about the number field.

3 Algebraic preliminaries

Our number field is $\mathbb{K} = \mathbb{Q}(\alpha) = \mathbb{Q}(\hat{\alpha})$, where α is an algebraic number and $\hat{\alpha} = c_d\alpha$ is an algebraic integer. Let $\mathcal{O}$ be its ring of integers, and $\mathcal{I}$ be the abelian group of fractional ideals of $\mathcal{O}$. For $x_1, \ldots, x_m \in \mathbb{K}$, we note $< x_1, \ldots, x_m >$ the element of $\mathcal{I}$ generated by $x_1, \ldots, x_m$. For every prime ideal $\mathfrak{p}$, we denote by $v_\mathfrak{p}$ the $\mathfrak{p}$-adic valuation that maps $\mathcal{I}$ to $\mathbb{Z}$. We define the *numerator* and *denominator* of $I \in \mathcal{I}$ to be the integral ideals $\text{numer}(I) = \prod_{v_\mathfrak{p}(I)>0} \mathfrak{p}^{v_\mathfrak{p}(I)}$ and $\text{denom}(I) = \prod_{v_\mathfrak{p}(I)<0} \mathfrak{p}^{-v_\mathfrak{p}(I)}$. We denote the norm of an ideal I by $\mathcal{N}(I)$, and the norm of an algebraic number $x \in \mathbb{K}$ by $\mathcal{N}_{\mathbb{K}}(x) = \prod_{1 \le i \le d} \sigma_i(x)$, σ_i denoting the d distinct embeddings of $\mathbb{K}$ in $\mathbb{C}$. We define the *complexity* of $I \in \mathcal{I}$ to be $\mathcal{C}(I) = \mathcal{N}(\mathrm{numer}(I))\mathcal{N}(\mathrm{denom}(I))$, and we say that I is *simpler* than J when $\mathcal{C}(I) \le \mathcal{C}(J)$. We say that a fractional ideal I is a *square* if there exists $J \in \mathcal{I}$ such that $J^2 = I$. Such a J is unique and will be denoted $\sqrt{I}$. If $\mathfrak{p}_1^{v_1} \dots \mathfrak{p}_m^{v_m}$ is the prime ideal factorization of I then: I is a square if and only if every v_i is even; if I is a square, then $\sqrt{I} = \mathfrak{p}_1^{v_1/2} \dots \mathfrak{p}_m^{v_m/2}$; if x is a square in $\mathbb{K}$, then so is $<x>$ in $\mathcal{I}$. We follow the notations of [3] and recall some results. Let $\mathcal{R}$ be an order in $\mathcal{O}$. By a "prime of $\mathcal{R}$" we mean a non-zero prime ideal of $\mathcal{R}$. We denote by $\{l_{\mathfrak{p},\mathcal{R}} : \mathbb{K}^* \to \mathbb{Z}\}_{\mathfrak{p}}$ the unique collection (where $\mathfrak{p}$ ranges over the set of all primes of $\mathcal{R}$) of group homomorphisms such that:

- $l_{\mathfrak{p},\mathcal{R}}(x) \ge 0$ for all $x \in \mathcal{R}, x \ne 0$;
- if x is a non-zero element of $\mathcal{R}$, then $l_{\mathfrak{p},\mathcal{R}}(x) > 0$ if and only if $x \in \mathfrak{p}$;
- for each $x \in \mathbb{K}^*$ one has $l_{\mathfrak{p},\mathcal{R}}(x) = 0$ for all but finitely many $\mathfrak{p}$, and $\prod_{\mathfrak{p}} \mathcal{N}(\mathfrak{p})^{l_{\mathfrak{p},\mathcal{R}}(x)} = |\mathcal{N}_{\mathbb{K}}(x)|$, where $\mathfrak{p}$ ranges over the set of all primes of $\mathcal{R}$.

$l_{\mathfrak{p},\mathcal{O}}(x)$ coincide with $v_{\mathfrak{p}}(<x>)$. Let $\beta_i = c_d\alpha^{d-1-i} + c_{d-1}\alpha^{d-2-i} + \dots + c_{i+1}$. We know that $\mathcal{A} = \mathbb{Z} + \sum_{i=0}^{d-2} \beta_i\mathbb{Z}$ is an order of $\mathcal{O}$, which is in fact $\mathbb{Z}[\alpha] \cap \mathbb{Z}[\alpha^{-1}]$. Its discriminant $\Delta(\mathcal{A})$ is equal to $\Delta(f)$ and we have:

$$\Delta(\mathbb{Z}[\hat{\alpha}]) = c_d^{(d-1)(d-2)}\Delta(\mathcal{A}), \qquad [\mathcal{O} : \mathbb{Z}[\hat{\alpha}]] = c_d^{\frac{(d-1)(d-2)}{2}}[\mathcal{O} : \mathcal{A}].$$

Recall that for any prime number p, $R(p)$ is defined as the set consisting of roots of f modulo p, together with ∞ if p divides c_d. Note that this $R(p)$ is denoted $R'(p)$ in [3]. The pairs consisting of a prime number p and an element $r \in R(p)$ are in bijective correspondence with the first degree primes $\mathfrak{p}$ of $\mathcal{A}$:

- if $r \ne \infty$ then $\mathfrak{p}$ is the intersection of $\mathcal{A}$ and the kernel of the ring homomorphism $\psi_{p,r} : \mathbb{Z}[\alpha] \to \mathbb{F}_p$ that sends α to r.
- if $r = \infty$ then $\mathfrak{p}$ is the intersection of $\mathcal{A}$ and the kernel of the ring homomorphism $\psi_{p,\infty} : \mathbb{Z}[\alpha^{-1}] \to \mathbb{F}_p$ that sends α^{-1} to 0.

Let p be a prime number, r an element of $R(p)$ and a, b be coprime integers. If $a \equiv br \pmod{p}$ and $r \ne \infty$, or if $b \equiv 0 \pmod{p}$ and $r = \infty$, we define $e_{p,r}(a,b) = v_p(F(a,b))$ where v_p denotes the ordinary p-adic valuation. Otherwise, we set $e_{p,r}(a,b) = 0$. We have $\mathcal{N}_{\mathbb{K}}(a - b\alpha) = \pm\frac{1}{c_d}\prod_{p,r} p^{e_{p,r}(a,b)}$, the product ranging over all pairs p, r with p prime and $r \in R(p)$. Furthermore, for any coprime integers a, b and any first degree prime $\mathfrak{p}$ of $\mathcal{A}$ corresponding to a pair $p, r \in R(p)$, we have:

$$l_{\mathfrak{p},\mathcal{A}}(a - b\alpha) = \begin{cases} e_{p,r}(a,b) & \text{if } r \ne \infty \\ e_{p,r}(a,b) - v_p(c_d) & \text{if } r = \infty \end{cases}$$

Theorem 1. *Let a and b be coprime integers, and p be a prime number. Let $\mathfrak{p}$ be a prime ideal of $\mathcal{O}$ above p such that $v_{\mathfrak{p}}(<a - b\alpha>) \ne 0$. If p does not divide $[\mathcal{O} : \mathcal{A}]$ then:*

1. *For every $r \in R(p)$, there is a unique prime ideal $\mathfrak{p}_r$ of $\mathcal{O}$ that lies over the first degree prime ideal $\mathfrak{q}_r$ of $\mathcal{A}$ corresponding to the pair p, r. $\mathfrak{p}_r$ is a first degree prime ideal, given by $\mathfrak{p}_r = <p, \beta_0 - \psi_{p,r}(\beta_0), \ldots, \beta_{d-2} - \psi_{p,r}(\beta_{d-2})>$. Furthermore, we have $v_{\mathfrak{p}_r}(<a - b\alpha>) = l_{\mathfrak{q}_r,\mathcal{A}}(a - b\alpha)$.*
2. *There is at most one finite $r \in R(p)$ such that $e_{p,r}(a, b) \neq 0$.*
3. *If p does not divide c_d, such a finite r exists and $\mathfrak{p} = \mathfrak{p}_r$.*
4. *If p divides c_d, then either $\mathfrak{p}$ is $\mathfrak{p}_\infty$, or $\mathfrak{p}_r$ for r finite.*
5. *p divides $F(a, b)$ or c_d.*

Proof. Let $r \in R(p)$ and $\mathfrak{q}_r$ be the first degree prime ideal of $\mathcal{A}$ corresponding to the pair p,r. Since p does not divide $[\mathcal{O} : \mathcal{A}]$, we have from [3] (Proposition 7.3, pages 65-66): $\sum_{\mathfrak{p}_r|\mathfrak{q}_r} f(\mathfrak{p}_r/\mathfrak{q}_r) = 1$, where $\mathfrak{p}_r$ ranges over all primes of $\mathcal{O}$ lying over $\mathfrak{q}_r$ and f denotes the residual degree. This proves that $\mathfrak{p}_r$ is unique and is a first degree prime ideal. From [3] (Proposition 7.2, page 65), we also have:

$$l_{\mathfrak{q}_r,\mathcal{A}}(a - b\alpha) = \sum_{\mathfrak{p}'|\mathfrak{q}_r} f(\mathfrak{p}'/\mathfrak{q}_r) l_{\mathfrak{p}',\mathcal{O}}(a - b\alpha) = l_{\mathfrak{p}_r,\mathcal{O}}(a - b\alpha).$$

Hence, $v_{\mathfrak{p}_r}(a - b\alpha) = l_{\mathfrak{q}_r,\mathcal{A}}(a - b\alpha)$. Moreover, we know a $\mathbb{Z}$-basis for any ideal $\mathfrak{q}_r$ of $\mathcal{A}$, namely $(p, \beta_0 - \psi_{p,r}(\beta_0), \ldots, \beta_{d-2} - \psi_{p,r}(\beta_{d-2}))$. Since $\mathfrak{p}_r$ lies over $\mathfrak{q}_r$, this $\mathbb{Z}$-basis is a system of $\mathcal{O}$-generators for $\mathfrak{p}_r$. We therefore proved 1. From the definition of β_i, one sees that $\beta_i = c_i\alpha^{-1} + c_{i-1}\alpha^{-2} + \cdots + c_0\alpha^{-i-1}$, which proves that $\psi_{p,\infty}(\beta_i) = 0$. This simplifies the formula when $r = \infty$. One obtains 2 from the definition of $e_{p,r}$. Denote by $\mathfrak{q}$ the intersection of $\mathfrak{p}$ and $\mathcal{A}$. $\mathfrak{q}$ is a prime of $\mathcal{A}$ and $\mathfrak{p}$ lies over $\mathfrak{q}$. We have $l_{\mathfrak{q},\mathcal{A}}(a - b\alpha) \neq 0$ since $v_{\mathfrak{p}}(a - b\alpha) \neq 0$. From [3] (page 89), this proves that $\mathfrak{q}$ is a first degree prime ideal of $\mathcal{A}$. Hence, there exists $r \in R(p)$ such that $\mathfrak{q} = \mathfrak{q}_r$. From 1, this proves that $\mathfrak{p} = \mathfrak{p}_r$. This r is finite or infinite, and if r is finite, it is the r of 2. This proves 3 and 4. From the formula expressing $l_{\mathfrak{q},\mathcal{A}}(a - b\alpha)$ in terms of $e_{p,r}(a, b)$, we obtain 5. □

4 The square root algorithm

We recall that we want to compute a square root of the algebraic number $\gamma = \prod_{i \in S}(a_i - b_i\alpha)$. The algorithm is split as follows:

1. Transform γ in order to make $<\gamma>$ simpler. The running time of the rest of the algorithm heuristically depends on $\mathcal{C}(<\gamma>)$.
2. Compute $\sqrt{<\gamma>}$ from the prime ideal factorization of $<\gamma>$ given by the prime factorization of each $F(a_i, b_i)$.
3. Approximate $\sqrt{\gamma}$ from $\sqrt{<\gamma>}$: using lattice reductions, construct a sequence of algebraic integers $\delta_1, \ldots, \delta_L$ in $\mathcal{O}$ and signs $s_1, \ldots, s_L$ in $\{\pm 1\}$ such that $\theta = \gamma \prod_{\ell=1}^{L} \delta_\ell^{-2s_\ell}$ is a "small" algebraic integer. θ can be thought as the square of the "guessing-error" in the approximation $\prod_{\ell=1}^{L} \delta_\ell^{s_\ell}$ of $\sqrt{\gamma}$.
4. Since γ is a square, so is θ. Compute $\sqrt{\theta}$ using brute-force method. One is able to explicitly write θ because θ is a "small" algebraic integer.

We thus obtain $\sqrt{\gamma}$ as a product of algebraic integers with exponents ± 1:

$$\sqrt{\gamma} = \sqrt{\theta} \prod_{\ell=1}^{L} \delta_\ell^{s_\ell}.$$

This enables to compute $\phi(\sqrt{\gamma})$ without explicitly calculating $\sqrt{\gamma}$, and hopefully some factors of n.

Although formalized differently, Montgomery's algorithm uses the same strategy. Only the steps change. We use another heuristic approach in Step 1, which seems to be more effective in practice. We use a new process in Step 2, derived from Section 3. Montgomery used a process which was as efficient, but only heuristic. Step 3 is the core of the algorithm. We modified this step by using the integral basis in a systematic manner, instead of the power basis. This simplifies the algorithm and the proofs. Heuristically, this should also improve the performances. We postpone the computation of the error in Step 4, while Montgomery included it in Step 3, by updating the computations during the approximation. This decreases the running-time because it is easier to estimate the necessary computations when Step 3 is over, and sometimes, Step 4 can be avoided (when the approximation is already perfect, which can be checked without additional computations). The new algorithm might be more suited to analysis, but like Montgomery's algorithm, its complexity has yet to be determined, even though they both display significantly better performances than former methods.

4.1 Computing in the number field

The ring of integers. During the whole algorithm, we need to work with ideals and algebraic integers. We first have to compute an integral basis of $\mathcal{O}$. In general, this is a hopeless task (see [13, 2] for a survey), but for the number fields NFS encounters (small degree and large discriminant), this can be done by the so-called round algorithms [16, 4]. Given an order $\mathcal{R}$ and several primes p_i, any round algorithm will enlarge this order for all these primes so that the new order $\widehat{\mathcal{R}}$ is p_i-maximal for every p_i. If we take for the p_i all the primes p such that p^2 divides $\Delta(\mathcal{R})$, then $\widehat{\mathcal{R}} = \mathcal{O}$. To determine all these primes, a partial factorization of $\Delta(\mathcal{R})$ suffices, that is a factorization of the form df^2 where d is squarefree and f is factorized. Theoretically, a partial factorization is as hard to find as a complete factorization and unfortunately, the discriminant is sometimes much larger than the number n we wish to factor. However, if one takes a "random" large number, and one removes all "small" prime factors from it (by trial division or by elliptic curves [12]), then in practice the result is quite likely to be squarefree. Furthermore, even in the case $\widehat{\mathcal{R}} \neq \mathcal{O}$, it will be true that $\widehat{\mathcal{R}}$ has almost all of the good properties of $\mathcal{O}$ for all ideals that we are likely to encounter in practice, like the fact that every ideal is a product of prime ideals. This is because every order satisfies these properties for all ideals that are coprime to the index of the order in $\mathcal{O}$. Hence, we can now assume that an integral basis $(\omega_1, \ldots, \omega_d)$ of $\mathcal{O}$ has been computed.

Algebraic numbers and ideals. From this integral basis we can represent any algebraic number of $\mathbb{K}$ as a vector of $\mathbb{Q}^d$: this is the integral representation. If $x \in \mathbb{K}$ we define $\boldsymbol{x} = [x_1, \ldots, x_d]^t$ where $x = \sum_{i=1}^{d} x_i \omega_i$ and $x_i \in \mathbb{Q}$. We can also represent any algebraic number as a polynomial of degree at most $d-1$ in α: this is the power representation. When dealing with algebraic integers, the integral representation is preferable. We will represent any integral ideal I by an integral matrix (with respect to $(\omega_1, \ldots, \omega_d)$) from a $\mathbb{Z}$-basis or a system of $\mathcal{O}$-generators. In the case of $\mathbb{Z}$-basis, we use the Hermite normal form (HNF) of the square matrix for efficiency reasons. We refer to [4] for algorithms concerning algebraic numbers and ideals.

4.2 Simplifying the principal ideal

If γ is a square in $\mathbb{K}$, then so is any $\gamma' = \prod_{i\in S}(a_i - b_i\alpha)^{e_i}$, when $e_i = \pm 1$. Since $\sqrt{\gamma} = \sqrt{\gamma'} \prod_{e_i=-1}(a_i - b_i\alpha)$, we can recover $\sqrt{\gamma}$ from $\sqrt{\gamma'}$ but actually, we only look for a square identity. Fortunately:

$$\left[\phi(\sqrt{\prod_{i\in S}(a_i - b_i\alpha)^{e_i}})\right]^2 \equiv \left[\sqrt{\prod_{i\in S}(a_i - b_i m)^{e_i}}\right]^2 \pmod n$$

This replaces the computation of $\sqrt{\gamma}$ by the computation of $\sqrt{\gamma'}$. By cleverly selecting the e_i, $\mathcal{C}(<\gamma'>)$ will be much smaller than $\mathcal{C}(<\gamma>)$: this is because many $<a_i - b_i\alpha>$ share the same prime ideals, since many $\mathcal{N}_{\mathbb{K}}(a_i - b_i\alpha)$ share the same primes (as a consequence of sieving). We now address the optimization problem of selecting the e_i so that $\mathcal{C}(<\gamma'>)$ is small. Given a distribution of e_i, the complexity of $<\gamma'>$ can be computed by the following formula (which comes from the known "factorization" of each $a_i - b_i\alpha$ into primes of $\mathcal{A}$):

$$\prod_{p,r\neq\infty} p^{\left|\sum_{i\in S} e_i e_{p,r}(a_i,b_i)\right|} \times \prod_{p|c_d} p^{\left|\sum_{i\in S} e_i [e_{p,\infty}(a_i,b_i) - v_p(c_d)]\right|}.$$

The simplest method is a random strategy which selects randomly $e_i = \pm 1$. Another method is a greedy strategy (used in [7]): at every step, select $e_i = \pm 1$ according to the best complexity (whether we put $a_i - b_i\alpha$ in the numerator or in the denominator). This behaves better than the random strategy. But the best method so far in practice is based on simulated annealing [18], a well-known probabilistic solution method in the field of combinatorial optimization. Here, the configuration space is $E = \{-1, +1\}^{|S|}$, and the energy function U maps any $\boldsymbol{e} = (e_1, \ldots, e_{|S|}) \in E$ to $\ln \mathcal{C}(<\gamma>)$ where γ corresponds to $\boldsymbol{e}$. For any $\boldsymbol{e} \in E$, we define its neighbourhood $\mathcal{V}(\boldsymbol{e}) = \{(e_1, \ldots, e_{i-1}, -e_i, e_{i+1}, \ldots, e_{|S|}) \mid i = 1, \ldots, |S|\}$. We try to minimize U by the following algorithm, which performances depend on three parameters Θ_i, Θ_f (initial and final temperatures) and τ:

- select randomly $\boldsymbol{e} \in E$ and set $\Theta \longleftarrow \Theta_i$.
- choose randomly $\boldsymbol{f} \in \mathcal{V}(\boldsymbol{e})$ and set $\Delta \longleftarrow U(\boldsymbol{f}) - U(\boldsymbol{e})$. If $\Delta > 0$, set $p \longleftarrow \exp(-\Delta/\Theta)$, otherwise set $p \longleftarrow 1$. Then set $\boldsymbol{e} \longleftarrow \boldsymbol{f}$ with probability p, and $\Theta \longleftarrow \Theta \times \tau$.

– repeat previous step if $\Theta > \Theta_f$.

Although this method behaves better in practice than previous methods, theoretical estimates can hardly be given.

4.3 Ideal square root

From now on, we forget about the initial γ and set $\gamma = \prod_{i\in S}(a_i - b_i\alpha)^{e_i}$. We wish to obtain $\sqrt{\gamma}$ as a product of ideals with exponents lying in $\mathbb{Z}$ (this ideal is too large to be represented as a single matrix). This can be done by factoring into prime ideals the fractional ideal $<\gamma> = <\prod_{i\in S}(a_i - b_i\alpha)^{e_i}>$. We simplify the problem to the factorization of any linear expression $<a_i - b_i\alpha>$ with coprime a_i, b_i. Such a factorization could be obtained by general ideal factorization algorithms (see [4]) but this would be too slow if we had to use these algorithms $|S|$ times. Fortunately, we can do much of the work by ourself using the known factorization of each $F(a_i, b_i) = f(a_i/b_i)b_i^d$, as shown in the previous section. We say that a prime number p is *exceptional* if p divides the index $\kappa = [\mathcal{O} : \mathcal{A}]$. Otherwise, we say that p is *normal.* Naturally, a prime ideal of $\mathcal{O}$ is said to be exceptional (resp. normal) if it lies above an exceptional (resp. normal) prime. If m is the number of prime factors of κ, there are at most md exceptional prime ideals. We compute all the exceptional prime ideals (for example, by decomposing all the exceptional primes in $\mathcal{O}$ using the Buchmann-Lenstra algorithm described in [4]), along with some constants allowing us to compute efficiently any valuation at these primes. From Theorem 1, we get the prime ideal factorization of $<a - b\alpha>$ as follows: for every prime number p dividing c_d or such that there exists a finite $r \in R(p)$ satisfying $e_{p,r}(a, b) \neq 0$,

– if p is exceptional, compute the valuation of $<a - b\alpha>$ at all the exceptional ideals lying above p.
– otherwise, p is normal. If there is a finite $r \in R(p)$ such that $e_{p,r}(a, b) \neq 0$ (r is then unique), pick the prime ideal $\mathfrak{p}_r$ with exponent $e_{p,r}(a, b)$ where

$$\mathfrak{p}_r = <p, \beta_0 - \psi_{p,r}(\beta_0), \ldots, \beta_{d-2} - \psi_{p,r}(\beta_{d-2})> \ .$$

  If $\infty \in R(p)$, also pick the prime ideal $\mathfrak{p}_\infty$ with exponent $e_{p,\infty}(a, b) - v_p(c_d)$ where $\mathfrak{p}_\infty = <p, \beta_0, \ldots, \beta_{d-2}>$.

We thus decompose $<\gamma>$ as a product of ideals where every exponent is necessarily even, which gives $\sqrt{<\gamma>}$. Montgomery used a different ideal factorization process (see [7, 14]) by introducing a special ideal, but its correctness is not proved.

4.4 Square root approximation

We now use the ideal square root $\sqrt{<\gamma>}$ to approximate $\sqrt{\gamma}$. Since $\sqrt{<\gamma>}$ is a huge ideal, we will get an approximation through an iterative process, by selecting a small part of the ideal at each step: this small part will be alternatively taken in the numerator and denominator. To lift an integral ideal to an algebraic integer, we use lattice reduction techniques. We associate several variables at each step ℓ:

- an algebraic number γ_ℓ. It can be considered as the square of the error in the current approximation of $\sqrt{\gamma}$.
- a sign s_ℓ in $\{-1,+1\}$, indicating whether we take something in the denominator or in the numerator of the huge original ideal.
- a fractional ideal G_ℓ, which is an approximation to $\sqrt{<\gamma_\ell>}$.
- an integral ideal H_ℓ of bounded norm. It differentiates G_ℓ from $\sqrt{<\gamma_\ell>}$.
- an algebraic integer δ_ℓ.
- an integral ideal I_ℓ of bounded norm.

We initialize these variables by: $\gamma_1 = \gamma = \prod_{i\in S}(a_i - b_i\alpha)^{e_i}$, $G_1 = \sqrt{<\gamma>}$, $H_1 = <1>$, $s_1 = 1$ if $\mathcal{N}_{\mathbb{K}}(\gamma) \geq 1$ and -1 otherwise. Each step of the approximation makes $\gamma_{\ell+1}$ in some sense smaller than γ_ℓ, and $G_{\ell+1}$ simpler than G_ℓ. After enough steps, G_ℓ is reduced to the unit ideal $<1>$, and γ_ℓ becomes an algebraic integer sufficiently small that its integral representation can be determined explicitly (using Chinese Remainders) and a square root constructed using brute-force method. At the start of step ℓ, we need to know the following:

- approximations to the $|\sigma_j(\gamma_\ell)|$ for $1 \leq j \leq d$, giving an approximation to $|\mathcal{N}_{\mathbb{K}}(\gamma_\ell)|$.
- prime ideal factorization of G_ℓ.
- Hermite normal form of H_ℓ.
- value of s_ℓ.

For $\ell = 1$, these information are obtained from the initial values of the variables. Each step ℓ consists of:

1. Select an integral ideal I_ℓ of almost fixed norm, by multiplying H_ℓ with another integral ideal dividing the numerator (resp. the denominator) of G_ℓ if $s_\ell = 1$ (resp. $s_\ell = -1$). Compute its Hermite normal form.
2. Pick some "nice" δ_ℓ in I_ℓ using lattice reductions.
3. Define:
$$\gamma_{\ell+1} = \gamma_\ell \delta_\ell^{-2s_\ell}, \quad G_{\ell+1} = \left(\frac{I_\ell}{H_\ell}\right)^{-s_\ell} G_\ell, \quad H_{\ell+1} = \frac{<\delta_\ell>}{I_\ell}, \quad s_{\ell+1} = -s_\ell.$$
   This allows to easily update necessary information:
   - compute the $|\sigma_j(\delta_\ell)|$'s to approximate the $|\sigma_j(\gamma_{\ell+1})|$'s.
   - the selection of I_ℓ is actually made in order to obtain the prime ideal factorization of $G_{\ell+1}$ simply by updating the exponents of the prime ideal factorization of G_ℓ.
   - $H_{\ell+1}$ and $s_{\ell+1}$ are directly computed.
4. Store s_ℓ and the integral representation of δ_ℓ.

We now explain the meaning of the different variables, then we detail the first two parts. By induction on ℓ, $\gamma = \gamma_\ell \left[\prod_{L=1}^{\ell-1} \delta_L^{s_L}\right]^2$. In other words, $\prod_{L=1}^{\ell-1} \delta_L^{s_L}$ is the approximation of $\sqrt{\gamma}$ at step ℓ. Each γ_ℓ is a square and $G_\ell = H_\ell^{s_\ell}\sqrt{<\gamma_\ell>}$. Notice that $\mathcal{C}(G_{\ell+1}) = \frac{1}{\mathcal{N}(I_\ell/H_\ell)}\mathcal{C}(G_\ell)$.

Ideal selection. We try to select an I_ℓ with norm as close as possible to a constant LLL_{max}, set at the beginning of the iterative process, to be explained later on. To do so, we adopt a greedy strategy. Since we know the prime ideal factorization of G_ℓ, we can sort all the prime ideals (according to their norm) appearing in this factorization. We start with $I_\ell = H_\ell$, and we keep multiplying I_ℓ by the possibly largest prime ideal power in such manner that $\mathcal{N}(I_\ell)$ is less than LLL_{max}. In practice, this strategy behaves well because most of our prime ideals lie over small primes. At the same time, when we pick a prime ideal power to multiply with I_ℓ, we update its exponent in the prime ideal factorization of G_ℓ so that we obtain the prime ideal factorization of $G_{\ell+1}$. At the end of the approximation, when $\mathcal{C}(G_\ell)$ is small, we find an I_ℓ of small norm (not close to LLL_{max}) such that $\frac{I_\ell}{H_\ell}$ equals the whole numerator or the whole denominator of G_ℓ.

Integer selection. We look for a nice element δ_ℓ in the integral ideal I_ℓ, that is to say, an algebraic integer that looks like the ideal. For us, "looking like" will mainly mean "with norm almost alike". This really means something since the norm of any element is a multiple of the norm of the integral ideal. So we select δ_ℓ in order to make $\mathcal{N}(<\delta_\ell> /I_\ell)$ as small as possible, which is the same as finding a short element in a given ideal. Fortunately an ideal is also a lattice, and there exists a famous polynomial-time algorithm for lattice reduction: LLL [9,4]. We will use two features of the LLL-algorithm: computation of an LLL-reduced basis, and computation of a short vector (with respect to the Euclidean norm, not to the norm in a number field). First, we reduce the basis of I_ℓ given by its HNF. In other words, we reduce the matrix of the integral representations (with respect to $(\omega_1, \ldots, \omega_d)$) of the elements of the basis. We do so because the HNF matrix is triangular, therefore not well-balanced: by applying an LLL reduction, coefficients are smaller and better spread.

Assume the obtained reduced basis is $(v^{(j)})_{j=1}^d$. We specify a constant $c > 0$ by

$$c^d = \frac{LLL_{max}}{\mathcal{N}(I_\ell)} \sqrt{\frac{|\mathcal{N}_{\mathbb{K}}(\gamma_\ell)|^{s_\ell}}{|\Delta(\mathbb{K})|}}.$$

Let $\lambda_j = \frac{c}{|\sigma_j(\gamma_\ell)|^{s_\ell/2}}$ for $1 \leq j \leq d$. We define a linear transformation Ω that maps any $v = \sum_{i=1}^d v_i \omega_i \in I_\ell$ to $\Omega v = [v_1, \ldots, v_d, \lambda_1 \sigma_1(v), \ldots, \lambda_d \sigma_d(v)]^t$. This is when $\mathbb{K}$ is totally real. If f has complex roots: for any complex conjugate pairs σ_i and $\overline{\sigma}_i$, we replace $\sigma_i(v)$ and $\overline{\sigma}_i(v)$ in the definition of Ω by respectively, $\Re(\sigma_i(v))\sqrt{2}$ and $\Im(\sigma_i(v))\sqrt{2}$. In Montgomery's implementation, the $\mathbb{Z}$-basis $(v^{(j)})_{j=1}^d$ is expressed with respect to the power basis instead of the integral basis, which does not seem to be more attractive. From $(v^{(j)})_{j=1}^d$, we form a $2d \times d$ real matrix with the corresponding $(\Omega v^{(j)})_{j=1}^d$.

Proposition 2. *This matrix satisfies:*

1. *The determinant of the image of the first d coordinates is in absolute value equal to $\mathcal{N}(I_\ell)$.*

2. *The determinant of the image of the last d coordinates is in absolute value equal to LLL_{max}.*

Proof. The image of the first d coordinates is the matrix representation of a $\mathbb{Z}$-basis of I_ℓ with respect to a $\mathbb{Z}$-basis of $\mathcal{O}$. Hence, its determinant is in absolute value equal to $[\mathcal{O} : I_\ell]$, proving 1. For 2, we assume that $\mathbb{K}$ is totally real: otherwise, the determinant is unchanged by multilinearity. In absolute value, the determinant of the image of the last d coordinates of $(\Omega v^{(j)})_{j=1}^{d}$ is equal to

$$\sqrt{|\Delta(v^{(1)}, \ldots, v^{(d)})|} \times \frac{c^d}{|\mathcal{N}_{\mathbb{K}}(\gamma_\ell)|^{s_\ell/2}},$$

where Δ denotes the discriminant of d elements of $\mathbb{K}$. Since the $v^{(j)}$ form a $\mathbb{Z}$-basis of I_ℓ, this discriminant is $\mathcal{N}(I_\ell)^2 \times \Delta(\omega_1, \ldots, \omega_d)$, where $\Delta(\omega_1, \ldots, \omega_d) = \Delta(\mathbb{K})$. The initial determinant is thus in absolute value $c^d |\mathcal{N}_{\mathbb{K}}(\gamma_\ell)|^{-s_\ell/2} \mathcal{N}(I_\ell) \sqrt{|\Delta(\mathbb{K})|}$, and we conclude from the definition of c. □

We apply a second LLL reduction to this matrix. In practice, we apply a LLL reduction to this matrix rounded to an integral matrix (notice that the upper $d \times d$ matrix has integral entries) as integer arithmetic is often preferable. We initialize LLL_{max} to the maximal value where the LLL-reduction algorithm supposedly performs well. The previous proposition ensures us that both LLL reductions perform well. We choose for δ_ℓ the algebraic integer defined by the first d coordinates of the first column of the matrix output by the second LLL reduction. We use the following result to prove that the approximation stage terminates.

Theorem 3. *There exists a computable constant C depending only on $\mathbb{K}$ such that the second LLL reduction outputs an algebraic integer δ_ℓ with*

$$|\mathcal{N}_{\mathbb{K}}(\delta_\ell)| \leq C \times \mathcal{N}(I_\ell),$$

where C is independent of $\mathcal{N}(I_\ell)$, LLL_{max} and c. In particular, $\mathcal{N}(H_\ell) \leq C$.

The proof, quite technical, is left in the appendix.

End of the approximation. We stop the iterative process when $\mathcal{C}(G_\ell) = 1$. This necessarily happens if $LLL_{max} \gg C$. Indeed, if numer$(\sqrt{<\gamma>})$ and denom$(\sqrt{<\gamma>})$ have close norms, then at every step ℓ, $\mathcal{N}(I_\ell/H_\ell)$ is close to LLL_{max}/C, which gives $\mathcal{C}(G_\ell) \approx (C/LLL_{max})^{\ell-1}\mathcal{C}(G_1)$. So the number of steps to obtain $\mathcal{C}(G_\ell) = 1$ is roughly logarithmic in $\mathcal{C}(\sqrt{<\gamma>})$. More precisely, one can show that if LLL_{max}/C is greater than the largest prime appearing in $\mathcal{C}(<\gamma>)$, then at most $2\lceil \log_2 \mathcal{C}(\sqrt{<\gamma>})\rceil$ steps are necessary to make $\mathcal{C}(G_\ell)$ equal to 1.

Once $\mathcal{C}(G_\ell) = 1$, we perform one more iteration if $s_\ell = +1$, in which $I_{\ell+1}$ is equal to H_ℓ. We can now assume that $\mathcal{C}(G_L) = 1$ with $s_L = -1$. This implies that $\sqrt{<\gamma_L>} = H_L$ and therefore, γ_L is an algebraic integer of norm $\mathcal{N}(H_L)^2$ bounded by C^2. This does not prove that γ_L has a small integral representation: if the coefficients of γ_L are small, then we can bound $\mathcal{N}_{\mathbb{K}}(\gamma_L)$, but the converse is false (for instance, γ_L might be a power of a unit).

Proposition 4. *There exists a computable constant C' depending only on $\mathbb{K}$ such that for every algebraic number $\theta = \sum_{j=1}^{d} \theta_j \omega_j \in \mathbb{K}$, each $|\theta_i|$ is bounded by*

$$C' \sqrt{\sum_{1 \le i \le d} |\sigma_i(\theta)|^2}.$$

Proof. Let Φ be the injective $\mathbb{Q}$-linear transformation that maps any $x \in \mathbb{K}$ to $[\sigma_1(x), \ldots, \sigma_d(x)]^t$. Since $\Phi(\mathbb{K})$ and $\mathbb{K}$ both are $\mathbb{Q}$-vector spaces of finite dimension, there exists $\|\Phi^{-1}\| \in \mathbb{R}$ such that for all $x \in \mathbb{K}$: $\|\boldsymbol{x}\| \le \|\Phi^{-1}\|.\|\Phi(x)\|$, where we consider the "Euclidean" norms induced on $\mathbb{K}$ by the integral basis $(\omega_1, \ldots, \omega_d)$, and on $\Phi(\mathbb{K})$ by the canonical basis of $\mathbb{C}^d$. The matrix $A = (\sigma_i(\omega_j))_{1 \le i,j \le d}$ represents Φ. A can be computed, and so can be its inverse A^{-1}. This gives an upper bound to $\|\Phi^{-1}\|$, which we note C'. □

With Lemma 5 (see the appendix), this proves that bounding the embeddings is the same as bounding the coefficients. But the linear transformation Ω is precisely chosen to reduce the embeddings: the last d coordinates reduce the sum of inverses of the embeddings of $\gamma_{\ell+1}$. This is not a proof, but it somehow explains why one obtains in practice a "small" algebraic integer.

4.5 Computing the error

We wish to compute the last algebraic integer $\theta = \gamma_L$ of norm at most C^2. We have a product formula for θ, of which we know every term. The partial products are too large to use directly this formula, but since we only deal with integers, we can use the Chinese Remainder Theorem if we choose good primes. A prime p is a *good prime* if it is inert (f is irreducible modulo p) and if p does not divide any of the $\mathcal{N}_{\mathbb{K}}(\delta_\ell)/\mathcal{N}(I_\ell)$. For such a p, the integral representation of $\theta \pmod p$ can be computed. This computation is not expensive if p is not too large. In general, it is easy to find good primes. We first find inert primes. In some very particular cases, inert primes do not even exist, but in general, there are a lot of inert primes (see [3]). Then we select among these primes those who do not divide any of the $\mathcal{N}_{\mathbb{K}}(\delta_\ell)/\mathcal{N}(I_\ell)$. Most of these primes will satisfy this assumption. If we selected several good primes $p_1, \ldots, p_N$, and if the coefficients of θ are all bounded by the product $p_1 \ldots p_N$, then we obtain these coefficients from the coefficients of θ modulo each p_i. In practice, a few good primes suffice. Then we can factorize $X^2 - \theta$ over $\mathbb{K}[X]$ in a reasonable time. The initial square root follows since $\sqrt{\gamma} = \sqrt{\theta} \prod_{\ell=1}^{L} \delta_\ell^{s_\ell}$. Actually, we only need $\phi(\sqrt{\gamma})$, so we compute all the $\phi(\delta_\ell)$ to avoid excessively large numbers. We thus obtain a square identity and hopefully, some factors of n.

5 Complexity analysis

We discuss the complexity of each stage of the algorithm, with respect to the growth of $|S|$. We assume that f is independent of $|S|$, which implies that all

a_i, b_i and $F(a_i, b_i)$ can be bounded independently of $|S|$. Recall that during the sieving, all $e_{p,r}(a, b)$ are computed.

Simplification of $<\gamma>$: even if the simulated annealing method is used, one can easily show that this stage takes at most $O(|S|)$ time.

Ideal square root: The only expensive operations are the decomposition of exceptional primes and the computation of valuations at these primes. The decomposition of exceptional primes is done once for all, independently of $|S|$. Any valuation can be efficiently computed, and takes time independent of $|S|$. Since exceptional prime numbers appear at most $O(|S|)$ times, this stage takes at most $O(|S|)$ time.

Square root approximation: We showed that the number of required steps was $O(\ln \mathcal{C}(<\gamma>))$. Since all the $F(a_i, b_i)$ are bounded, $\ln \mathcal{C}(<\gamma>)$ is $O(|S|)$. Unfortunately, we cannot say much about the complexity of each step, although each step takes very little time in practice. This is because we cannot bound independently of $|S|$ all the entries of the $2d \times d$ matrix that is LLL reduced. Indeed, we can bound the entries of the upper $d \times d$ square matrix, but not the entries of the lower one, as we are unable to prove that the embeddings of the algebraic number γ_ℓ get better. However, since we perform LLL reductions on matrices with very small dimension, it is likely that these reductions take very little time, unless the entries are extremely large. This is why in practice the approximation takes at most $O(|S|)$ time.

Computing the error: If we can bound the number and the size of necessary good primes independently of $|S|$, then this stage takes at most $O(|S|)$ time. Unfortunately, we are unable to do this, because we cannot bound the embeddings of the last algebraic integer θ, as seen previously. In practice however, these embeddings are small.

One sees that it is difficult to prove anything on the complexity of the algorithm. The same holds for Montgomery's algorithm. In practice, the algorithm behaves as if it had linear time in $|S|$ (which is not too surprising), but we are unable to prove it at the moment. We lack a proof mainly because we do not know any particular expression for $\sqrt{\gamma}$. For instance, we do not know if $\sqrt{\gamma}$ can be expressed as a product with exponents ± 1 of algebraic integers with bounded integral representation.

6 Implementation

We make some remarks about the implementation:

1. Since the number of ideals appearing in $\sqrt{<\gamma>}$ is huge, we use a hash-table and represent any normal prime ideal by its corresponding (p, r) pair. Exceptional prime ideals require more place, but there are very few exceptional primes.
2. It is only during the approximation process (namely, to obtain the Hermite normal form of I_ℓ) that one needs to compute a system of $\mathcal{O}$-generators for normal prime ideals. Such a computation is however very fast.

3. To avoid overflows, we do not compute $|\sigma_j(\gamma_\ell)|$, c and λ_j but their logarithms. One checks that $\sum_{j=1}^{d} \ln|\sigma_j(\gamma_\ell)| = \ln|\mathcal{N}_{\mathbb{K}}(\gamma_\ell)|$ if one is in doubt about the precision.
4. To choose the constant LLL_{max}, one can compute the C constant from the formulas given in the proof of Theorem 3, but one can also perform some LLL reductions to obtain the practical value of C. Notice that when one knows C and LLL_{max}, one can estimate the number of iterations.
5. To know how many good primes are sufficient to compute the last algebraic integer, one can compute the C' constant as shown in the proof of Proposition 4, which gives a bound for the coefficients of the integral representation.
6. The last algebraic integer is often a small root of unity. This is because the last ideal I_ℓ is principal, and we know an approximation to the embeddings of one of its generators. This generator has unusual short norm in the corresponding lattice, therefore it is no surprise that the LLL algorithm finds this generator, making $H_{\ell+1}$ equal to $<1>$. In the latter case, the last algebraic integer is often equal to ± 1: one should try to bypass the computation of the error and apply ϕ directly to find some factors of n.

The algorithm has been implemented using version 1.39 of the PARI library [1] developed by Henri Cohen et al. In December, 1996, it completed the factorization of the 100-digit cofactor of $17^{186}+1$, using the quadratic polynomials $5633687910X^2 - 4024812630168572920172347X + 48297751562022581583320305619782859 1062$ and $-77869128383X^2 - 2888634446047190834964717X + 346636133525639208946167278118238554489$. Each dependency had about 1.5 million relations. It took the square root code about 10 hours to do both square roots on a 75Mhz Sparc 20.

7 Conclusion

We presented an algorithm suitable for implementation to solve the square root problem of the number field sieve. This algorithm is a variant of Montgomery's square root. We modified the square root approximation process by using an integral basis instead of the power basis: this allows to work with integers instead of rationals, and to search the algebraic integer δ_ℓ in the whole ideal I_ℓ, not in some of its submodules. We introduced the simulated annealing method in the ideal simplification process. From results of [3], we proposed an efficient ideal square root process and proved its validity. We postponed the computation of the error to avoid useless computations. The present running time of the algorithm is negligible compared to other stages of the number field sieve. In practice, the algorithm behaves as if it had linear complexity, but one should note that this is only heuristic as few things are proved about the complexity. It is an open problem to determine precisely the complexity of the algorithm.

Acknowledgements. I am particularly grateful to both Arjen and Hendrik Lenstra for many explanations about the number field sieve. I wish to thank Jean-Marc Couveignes and Peter Montgomery for enlightening discussions. I also thank Philippe Hoogvorst for his helpful comments, and for carrying out experiments.

A Proof of Theorem 3

This theorem is related to the classical result of the geometry of numbers which states that for any integral ideal I, there exists an algebraic integer $\delta \in I$ such that $|\mathcal{N}_{\mathbb{K}}(\delta)| \leq \mathcal{M}(\mathbb{K})\mathcal{N}(I)$ where $\mathcal{M}(\mathbb{K})$ denotes the Minkowski constant of $\mathbb{K}$. It relies on Minkowski's convex body theorem which can be viewed as a generalization of the pigeon-hole principle. Following an idea of Montgomery [14], we use the pigeon-hole principle to estimate precisely each component of δ_ℓ. The only thing we need to know about LLL-reduced bases is that if $(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_d)$ is an LLL-reduced basis of a lattice Λ, then

$$\det(\Lambda) \leq \prod_{i=1}^{d} \|\boldsymbol{b}_i\| \leq 2^{d(d-1)/4} \det(\Lambda) \tag{1}$$

$$\|\boldsymbol{b}_1\| \leq 2^{(d-1)/2} \|\boldsymbol{x}\| \quad \text{if} \quad \boldsymbol{x} \in \Lambda, \boldsymbol{x} \neq 0 \tag{2}$$

where *det* denotes the lattice determinant and $\|.\|$ denotes the Euclidean norm. In the following, we will use the notation $\|.\|$ even for vectors with different number of coordinates. Here, if $x = \sum_{i=1}^{d} x_i \omega_i$ is an algebraic number of $\mathbb{K}$, then $\|\boldsymbol{x}\| = \sqrt{\sum_{i=1}^{d} x_i^2}$. We will use the notation $(\boldsymbol{x})_i$ to denote the i-th coordinate of $\boldsymbol{x}$. From now on (all along the proof), we assume that $\mathbb{K}$ is totally real to simplify the definition of Ω, but a similar reasoning applies to other cases with a different choice of constants.

Lemma 5. *There exists a computable constant C_1 depending only on $\mathbb{K}$ such that for every $x \in \mathbb{K}$, and for any integer $j = 1, \ldots, d$:*

$$|\sigma_j(x)| \leq C_1 \|\boldsymbol{x}\| \tag{3}$$

$$|(\Omega x)_{d+j}| \leq \lambda_j C_1 \|\boldsymbol{x}\| \tag{4}$$

Proof. We have $x = \sum_{i=1}^{d} x_i \omega_i$ where $x_i \in \mathbb{Q}$. Therefore $\sigma_j(x) = \sum_{i=1}^{d} x_i \sigma_j(\omega_i)$. Using triangle inequality and Cauchy-Schwarz, we obtain:

$$|\sigma_j(x)| \leq \sum_{i=1}^{d} |x_i||\sigma_j(\omega_i)| \leq \sqrt{\sum_{i=1}^{d} |x_i|^2} \times \sqrt{\sum_{i=1}^{d} |\sigma_j(\omega_i)|^2} \leq \|\boldsymbol{x}\| C_1,$$

where $C_1 = \max_{1 \leq j \leq d} \sqrt{\sum_{i=1}^{d} |\sigma_j(\omega_i)|^2}$. This proves (3), which implies (4) by definition of Ω. $\square$

Lemma 6. *There exists two computable constants C_2 and C_3 depending only on $\mathbb{K}$ such that for any integral ideal I_ℓ, there exists a real M and an algebraic* *integer $z \in I_\ell, z \neq 0$ satisfying:*

$$M^d \leq C_2 \prod_{j \in J} \lambda_j \tag{5}$$

$$\|\boldsymbol{z}\| \leq M\mathcal{N}(I_\ell)^{1/d} \tag{6}$$

$$\forall j \in J \quad \lambda_j \|\boldsymbol{z}\| \leq M\mathcal{N}(I_\ell)^{1/d} \tag{7}$$

$$\|\Omega z\| \leq C_3 M\mathcal{N}(I_\ell)^{1/d} \tag{8}$$

where $J = \{j = 1, \ldots, d \ / \ \lambda_j > 1\}$.

Proof. Let $C_2 = 2^{d(d-1)/4} d^d 2^{d+1}$. Since $2^{d(d-1)/4} d^d \prod_{j \in J} \lceil \lambda_j \rceil < C_2 \prod_{j \in J} \lambda_j$ by definition of J, there exists $M > 0$ such that $2^{d(d-1)/4} d^d \prod_{j \in J} \lceil \lambda_j \rceil < M^d \leq C_2 \prod_{j \in J} \lambda_j$. This M satisfies (5). The number of $n = (n_1, \ldots, n_d) \in \mathbb{N}^d$ such that each n_i satisfies $n_i \|\boldsymbol{v}^{(i)}\| \leq \frac{M}{d} \mathcal{N}(I_\ell)^{1/d}$ is at least

$$\prod_{i=1}^{d} \lceil \frac{M\mathcal{N}(I_\ell)^{1/d}}{d\|\boldsymbol{v}^{(i)}\|} \rceil \geq \prod_{i=1}^{d} \frac{M\mathcal{N}(I_\ell)^{1/d}}{d\|\boldsymbol{v}^{(i)}\|} \geq \frac{M^d}{d^d 2^{d(d-1)/4}} \text{ by (1)} > \prod_{j \in J} \lceil \lambda_j \rceil.$$

For such an n, $\lfloor \lambda_j \frac{n_i d \|\boldsymbol{v}^{(i)}\|}{M\mathcal{N}(I_\ell)^{1/d}} \rfloor$ is a positive integer less than λ_j. By the pigeonhole principle, there therefore exists two distinct $n = (n_1, \ldots, n_d)$ and $n' = (n'_1, \ldots, n'_d)$ both in $\mathbb{N}^d$ such that for all $i = 1, \ldots, d$:

$$n_i \|\boldsymbol{v}^{(i)}\| \leq \frac{M}{d} \mathcal{N}(I_\ell)^{1/d} \tag{9}$$

$$n'_i \|\boldsymbol{v}^{(i)}\| \leq \frac{M}{d} \mathcal{N}(I_\ell)^{1/d} \tag{10}$$

$$\forall j \in J \quad \lfloor \lambda_j \frac{n_i d \|\boldsymbol{v}^{(i)}\|}{M\mathcal{N}(I_\ell)^{1/d}} \rfloor = \lfloor \lambda_j \frac{n'_i d \|\boldsymbol{v}^{(i)}\|}{M\mathcal{N}(I_\ell)^{1/d}} \rfloor \tag{11}$$

Define $z = \sum_{i=1}^{d} (n_i - n'_i) v^{(i)}$. Then $z \in I_\ell, z \neq 0$ and by (9) and (10), we have for all $i = 1, \ldots, d$:

$$|n_i - n'_i| . \|\boldsymbol{v}^{(i)}\| \leq \frac{M}{d} \mathcal{N}(I_\ell)^{1/d}.$$

This proves (6) by triangle inequality . Furthermore, for all $j \in J$ and for all $i = 1, \ldots, d$, the quantity $\lambda_j |n_i - n'_i| . \|\boldsymbol{v}^{(i)}\|$ is equal to

$$\frac{M}{d} \mathcal{N}(I_\ell)^{1/d} \left| \lambda_j \frac{n_i d \|\boldsymbol{v}^{(i)}\|}{M\mathcal{N}(I_\ell)^{1/d}} - \lambda_j \frac{n'_i d \|\boldsymbol{v}^{(i)}\|}{M\mathcal{N}(I_\ell)^{1/d}} \right|,$$

which is, by (11), less than $\frac{M}{d}\mathcal{N}(I_\ell)^{1/d}$. This proves (7) by triangle inequality. Finally:

$$\begin{aligned}\|\Omega z\|^2 &= \sum_{j=1}^{d} |(\Omega z)_j|^2 + \sum_{j=1}^{d} |(\Omega z)_{d+j}|^2 \\ &\leq \|\boldsymbol{z}\|^2 + \sum_{j\notin J} \lambda_j C_1 \|\boldsymbol{z}\|^2 + \sum_{j\in J} \lambda_j C_1 \|\boldsymbol{z}\|^2 \text{ by (4)} \\ &\leq \left(1 + C_1 \sum_{j\notin J} 1 + C_1 \sum_{j\in J} 1\right) \left[M\mathcal{N}(I_\ell)^{1/d}\right]^2\end{aligned}$$

by (6), (7) and the definition of J. This proves (8) with $C_3 = \sqrt{1 + dC_1}$. □

Now, if δ is the algebraic integer output by the second LLL reduction, (2) implies that $\|\Omega\delta\|^2 \leq 2^{d-1}\|\Omega z\|^2$. Since $\|\boldsymbol{\delta}\| \leq \|\Omega\delta\|$, (8) implies that

$$\|\boldsymbol{\delta}\| \leq 2^{(d-1)/2} C_3 M \mathcal{N}(I_\ell)^{1/d}.$$

Moreover, $|\mathcal{N}_{\mathbb{K}}(\delta)| = \prod_{j=1}^{d} |\sigma_j(\delta)| = \left(\prod_{j\in J} |\sigma_j(\delta)|\right) \times \left(\prod_{j\notin J} |\sigma_j(\delta)|\right)$. On the one hand, by (3):

$$\prod_{j\notin J} |\sigma_j(\delta)| \leq (C_1\|\boldsymbol{\delta}\|)^{d-|J|} \leq \left[2^{(d-1)/2} C_1 C_3 M \mathcal{N}(I_\ell)^{1/d}\right]^{d-|J|}.$$

On the other hand, $\prod_{j\in J} |\sigma_j(\delta)| = \frac{\prod_{j\in J} |(\Omega\delta)_{d+j}|}{\prod_{j\in J} \lambda_j}$, where by the arithmetic-geometric mean inequality:

$$\begin{aligned}\prod_{j\in J} |(\Omega\delta)_{d+j}|^2 &\leq \left(\sum_{j\in J} |(\Omega\delta)_{d+j}|^2\right)^{|J|} \leq (\|\Omega\delta\|^2)^{|J|} \leq (2^{d-1}\|\Omega z\|^2)^{|J|} \\ &\leq \left[2^{(d-1)/2} C_3 M \mathcal{N}(I_\ell)^{1/d}\right]^{|J|} \text{ by (8).}\end{aligned}$$

We collect these two inequalities:

$$\begin{aligned}|\mathcal{N}_{\mathbb{K}}(\delta)| &\leq \frac{C_1^{d-|J|}}{\prod_{j\in J} \lambda_j} \left[2^{(d-1)/2} C_3 M \mathcal{N}(I_\ell)^{1/d}\right]^{d-|J|+|J|} \\ &\leq \frac{\max(1, C_1^d)}{\prod_{j\in J} \lambda_j} 2^{d(d-1)/2} C_3^d M^d \mathcal{N}(I_\ell) \\ &\leq \max(1, C_1^d) 2^{d(d-1)/2} C_3^d C_2 \mathcal{N}(I_\ell) \text{ by (5).}\end{aligned}$$

This completes the proof with $C = 2^{d(d-1)/2} \max(1, C_1^d) C_2 C_3^d$.

References

1. Batut, C., Bernardi, D., Cohen, H., and Olivier, M. Pari-gp computer package. Can be obtained by ftp at megrez.math.u-bordeaux.fr.
2. Buchmann, J. A., and Lenstra, Jr., H. W. Approximating rings of integers in number fields. *J. Théor. Nombres Bordeaux 6*, 2 (1994), 221–260.
3. Buhler, J. P., Lenstra, H. W., and Pomerance, C. Factoring integers with the number field sieve. pages 50-94 in [8].
4. Cohen, H. *A course in computational algebraic number theory.* Springer, 1993.
5. Couveignes, J.-M. Computing a square root for the number field sieve. pages 95-102 in [8].
6. Cowie, J., Dodson, B., Elkenbracht-Huizing, R. M., Lenstra, A. K., Montgomery, P. L., and Zayer, J. A world wide number field sieve factoring record: On to 512 bits. In *Proceedings of ASIACRYPT'96* (1996), vol. 1163 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 382–394.
7. Elkenbracht-Huizing, M. An implementation of the number field sieve. *Experimental Mathematics 5*, 3 (1996), 231–253.
8. Lenstra, A. K., and Lenstra, Jr., H. W. *The development of the Number Field Sieve*, vol. 1554 of *Lecture Notes in Mathematics.* Springer-Verlag, 1993.
9. Lenstra, A. K., Lenstra, Jr., H. W., and Lovász, L. Factoring polynomials with rational coefficients. *Math. Ann. 261* (1982), 515–534.
10. Lenstra, A. K., Lenstra, Jr., H. W., Manasse, M. S., and Pollard, J. M. The number field sieve. pages 11-42 in [8].
11. Lenstra, A. K., Lenstra, Jr., H. W., Manasse, M. S., and Pollard, J. M. The factorization of the ninth fermat number. *Math. Comp. 61* (1993), 319–349.
12. Lenstra, Jr., H. W. Factoring integers with elliptic curves. *Ann. of Math. 126* (1987), 649–673.
13. Lenstra, Jr., H. W. Algorithms in algebraic number theory. *Bull. Amer. Math. Soc. 26* (1992), 211–244.
14. Montgomery, P. L. Square roots of products of algebraic numbers. Draft of June, 1995. Available at `ftp://ftp.cwi.nl/pub/pmontgom/sqrt.ps.gz`.
15. Montgomery, P. L. Square roots of products of algebraic numbers. In *Mathematics of Computation 1943-1993: a Half-Century of Computational Mathematics* (1994), W. Gautschi, Ed., Proceedings of Symposia in Applied Mathematics, American Mathematical Society, pp. 567–571.
16. Pohst, M., and Zassenhaus, H. *Algorithmic algebraic number theory.* Cambridge University Press, 1989.
17. Pollard, J. M. Factoring with cubic integers. pages 4-11 in [8].
18. Reeves, C. R. *Modern Heuristic Techniques for Combinatorial Problems.* Blackwell Scientific Publications, 1993.