A variant of Coppersmith's Algorithm with Improved Complexity and Efficient Exhaustive Search

Jean-Sébastien Coron[1], Jean-Charles Faugère[2], Guénaël Renault[2], and Rina Zeitoun[2,3]

[1] University of Luxembourg
jean-sebastien.coron@uni.lu

[2] UPMC, Université Paris 6, INRIA, Centre Paris-Rocquencourt,
PolSys Project-team, CNRS, UMR 7606, LIP6
4 place Jussieu, 75252 Paris, Cedex 05, France
jean-charles.faugere@inria.fr
guenael.renault@lip6.fr

[3] Oberthur Technologies
420 rue d'Estienne d'Orves, CS 40008, 92705 Colombes, France
r.zeitoun@oberthur.com

Abstract. Coppersmith described at Eurocrypt 96 a polynomial-time algorithm for finding small roots of univariate modular equations, based on lattice reduction. In this paper we describe the first improvement of the asymptotic complexity of Coppersmith's algorithm. Our method consists in taking advantage of Coppersmith's matrix structure, in order to apply LLL algorithm on a matrix whose elements are smaller than those of Coppersmith's original matrix. Using the L^2 algorithm, the asymptotic complexity of our method is $\mathcal{O}(\log^{6+\varepsilon} N)$ for any $\varepsilon > 0$, instead of $\mathcal{O}(\log^{8+\varepsilon} N)$ previously. Furthermore, we devise a method that allows to speed up the exhaustive search which is usually performed to reach Coppersmith's theoretical bound. Our approach takes advantage of the LLL performed to test one guess, to reduce complexity of the LLL performed for the next guess. Experimental results confirm that it leads to a considerable performance improvement.

Keywords: Coppersmith's Method, LLL, Structured Matrix.

1 Introduction

The famous Diophantine problem of finding integer roots of polynomial equations is known to be hard in general. In cryptology, many security assumptions are based on the ability to solve specific diophantine equations. At Eurocrypt 96, Coppersmith used lattice reduction to design a method that allows to retrieve small integer roots [1], with many applications in cryptology, especially regarding RSA. Namely, for polynomials of the form: $p(x) = x^\delta + a_{\delta-1}x^{\delta-1}+...+a_0 \equiv 0 \mod N$ where N has an unknown factorization, Coppersmith proved that one can find any root x_0 satisfying $|x_0| < N^{1/\delta}$ in polynomial time. The technique he designed was latter simplified by Howgrave-Graham in [2]. Both methods have the same asymptotical complexity, but since the latter one holds a more natural approach and is easier to implement, it is commonly adopted. The method consists in building a lattice which contains polynomials that admit x_0 as a root modulo N^ℓ, where ℓ is a parameter related to the dimension of the lattice. This lattice is then reduced using the well-known LLL algorithm [3] or an analogous algorithm with improved complexity [4, 5]. Since the first resulting polynomial $h(x_0) \equiv 0 \mod N^\ell$ is provided with small coefficients, it is such that $h(x_0)$ equals to zero over the integers. The solution x_0 can therefore be retrieved by solving $h(x)$ using a classical method for factoring polynomials over finite fields (*e.g.* Berlekamp's algorithm). The complexity of the method depends almost exclusively on the LLL complexity. Namely, if we use the L^2 algorithm which is an improved version of the LLL algorithm due to Nguyen and Stehlé [4], the complexity of the method is $\mathcal{O}(\omega^{5+\varepsilon}\log B + \omega^{4+\varepsilon}\log^2 B)$ where ω is the dimension of the lattice and the bit-size of the elements is bounded by $\log B = \mathcal{O}(\log^2 N)$.

Coppersmith's method brought forth many applications in cryptology, especially regarding RSA cryptosystem. Namely, it has been used to factorize $N = pq$ with partial knowledge of p [6] or to factorize modulus of the form $N = p^r q$ with large r [7]. It has also been used for cryptanalysis of RSA with fixed pattern padding [8], with small secret CRT-exponents [9] or with $d < N^{0.29}$ [10].

As mentioned previously, the success of Coppersmith's methods in [1] and [2], depends on the size of the solution. More precisely, the largest the dimension of the lattice, the closest to $N^{1/\delta}$ the solution x_0 can be. In order to reach the bound $|x_0| < N^{1/\delta}$, one would need to use a lattice of huge dimension. Thus, the computation would become prohibitive. Therefore in practice, one uses lattices of reasonable dimension, which allows to retrieve most part of the solution x_0. Exhaustive search is performed to retrieve the other part α_0. The exhaustive search consists in testing all possible polynomials, each polynomial corresponding to one possible guess for α_0. For each guess, one builds a lattice associated to the polynomial according to [2], and one applies LLL to reduce it. The solution x_0 will be found for the right guess α_0, *i.e.* for the correct polynomial.

In this paper, we propose a variant of Coppersmith's approach that allows to improve the asymptotical complexity of the method. Our approach consists in applying LLL on a matrix containing smaller elements than in the original matrix. This is done by taking the integer quotient of all elements by a particular value. Then, one translates the obtained information back into the original matrix. This allows to obtain an LLL-reduced matrix at reduced cost, namely the bit-size of the elements is bounded by $\log B = \mathcal{O}(\log N)$ instead of $\log B = \mathcal{O}(\log^2 N)$.
We further exhibit a new approach to carry out the exhaustive search which consists in taking advantage of the LLL performed to test one guess, to reduce complexity of the LLL performed for the next guess. Namely, we show that matrices associated to both guesses are linked by a specific transformation, which allows to report the obtained information from one to the other. We also combine this method with the previous one, by truncating matrices before applying LLL. This new approach is heuristic and provides a considerable speed up. Namely, for polynomials of the form $p(x) = x^2 + ax + b \equiv 0 \mod N$ and $\lceil \log_2(N) \rceil = 2048$ the new method performs more than 1000 times faster than the original one.

The paper is organized as follows: Section 2 gives an overview of the main principles in Coppersmith's algorithm (we use Howgrave-Graham's variant [2]). Section 3 describes a method to reduce complexity of the LLL computation performed in [2]. A new heuristic approach to carry out exhaustive search is exhibited in Section 4. Experimental results are presented in Section 5. They validate the efficiency of both improvements. Eventually, in Section 6, we conclude our research.

2 Coppersmith's Algorithm

We first recall Coppersmith's algorithm. We use the Howgrave-Graham variant [2] which is equivalent to Coppersmith's method [1] since it has the same complexity as Coppersmith's method and has a more natural approach. Thus, in the sequel, by "Coppersmith's matrix", we mean the matrix depicted in [2].

Theorem 1 (Coppersmith). *Let $p(x)$ be a monic polynomial of degree δ in one variable modulo an integer N of unknown factorization. Let X be such that $X \leq N^{1/\delta}$. One can find all integers x_0 with $p(x_0) = 0 \bmod N$ and $|x_0| \leq X$, in time polynomial in* $(\log N, 2^\delta)$.

The method is depicted hereafter. Given a parameter $\ell \geq 1$, we consider the polynomials:

$$q_{ik}(x) = x^i \cdot N^{\ell-k} p^k(x) \bmod N^\ell$$

with $0 \leq i < \delta$ for all $0 \leq k < \ell$, and $i = 0$ for $k = \ell$. We have:

$$q_{ik}(x_0) = 0 \bmod N^\ell\,.$$

Let $h(x)$ be a linear combination of the $q_{ik}(x)$. Thusly, $h(x)$ is such that $h(x_0) = 0 \bmod N^\ell$. The core idea of the method consists in noting that if the solution x_0 and the coefficient of h(x) are sufficiently small, *i.e.* they are such that $h(x_0) < N^\ell$, then $h(x_0)$ holds over the integers and it can be solved using a classical method for factoring polynomials over finite fields (*e.g.* Berlekamp's algorithm). Beforehand, we need to recall the following lemma.

Lemma 1 (Howgrave-Graham). *Let $h(x) \in \mathbb{Z}[x]$ be the sum of at most ω monomials. Assume that $h(x_0) = 0 \bmod N^\ell$ where $|x_0| \leq X$ and $\|h(xX)\| < N^\ell/\sqrt{\omega}$. Then $h(x_0) = 0$ over the integers.*

Proof. We have:

$$\begin{aligned} |h(x_0)| &= \left|\sum h_i x_0^i\right| = \left|\sum h_i X^i \left(\frac{x_0}{X}\right)^i\right| \\ &\leq \sum \left|h_i X^i \left(\frac{x_0}{X}\right)^i\right| \leq \sum \left|h_i X^i\right| \\ &\leq \sqrt{\omega}\|h(xX)\| < N^\ell\,. \end{aligned}$$

Since $h(x_0) = 0 \bmod N^\ell$, this gives $h(x_0) = 0$. □

We consider the matrix M of dimension

$$\omega = \delta \cdot \ell + 1$$

whose row vectors are the coefficients of the polynomials $q_{ik}(xX)$. For $p(x) = x^2 + ax + b$ and $\ell = 1$, we have:

$$M = \begin{bmatrix} N & 0 & 0 \\ 0 & NX & 0 \\ b & aX & X^2 \end{bmatrix}.$$

The matrix M is triangular and diagonal elements of the matrix M are given by:

$$X^{\delta \cdot k + i} \cdot N^{\ell - k}$$

for $0 \leq i < \delta$ for all $0 \leq k < \ell$, and $i = 0$ for $k = \ell$. Therefore the determinant of the matrix is given by:

$$\det M = X^{\omega \cdot (\omega-1)/2} \cdot N^{\delta \cdot \ell \cdot (\ell+1)/2}\,.$$

We acknowledge the following theorem:

Theorem 2 (LLL). *Let L be a lattice spanned by $(u_1, \ldots, u_\omega) \in \mathbb{Z}^n$, where the Euclidean norm of each vector is bounded by B. The LLL algorithm, given $(u_1, \ldots, u_\omega)$, finds in time $\mathcal{O}(\omega^5 n \log^3 B)$ a vector b_1 such that:*

$$\|b_1\| \leq 2^{(\omega-1)/4} \det(L)^{1/\omega} .$$

Remark 1. In order to obtain a better complexity, one can use enhanced versions of LLL which achieve the same bound on $\|b_1\|$ but with improved complexity. Namely, the L^2 algorithm due to Nguyen and Stehlé [4], terminates in time $\mathcal{O}(\omega^{5+\varepsilon} \log B + \omega^{4+\varepsilon} \log^2 B)$ and the $\tilde{L}^1$ algorithm from Novocin, Stehlé and Villard [5] has a complexity $\mathcal{O}(\omega^{5+\varepsilon} \log B + \omega^{1+u+\varepsilon} \log^{1+\varepsilon} B)$ where $u = 2.376$ is a valid exponent for matrix multiplication.

Coppersmith's method consists in applying the LLL algorithm (or an equivalent) on the matrix M with dimension $\omega = \delta \cdot \ell + 1$. According to Theorem 2, the first resulting polynomial $h(xX)$ is such that:

$$\|h(xX)\| \leq 2^{(\omega-1)/4} (\det M)^{1/\omega} = 2^{(\omega-1)/4} X^{(\omega-1)/2} N^{\delta \cdot \ell \cdot (\ell+1)/(2\delta\ell+2)} .$$

In order to fulfill the condition $\|h(xX)\| < N^\ell/\sqrt{\omega}$ and neglecting the term $\sqrt{\omega}$ and $2^{(\omega-1)/4}$, we have the following inequality which yields a condition on the bound X:

$$X^{\delta \cdot \ell/2} N^{\delta \cdot \ell \cdot (\ell+1)/(2\delta\ell+2)} < N^\ell$$

which gives:

$$X^{\delta \cdot \ell/2} < N^{(\delta \cdot \ell^2 + 2\ell - \delta \cdot \ell)/(2\delta\ell+2)} .$$

Therefore we get the following upper-bound on X:

$$X < N^{(\delta \cdot \ell + 2 - \delta)/(\delta \cdot (\delta \cdot \ell + 1))} .$$

More precisely we get:

$$X < N^{1/\delta - \varepsilon} ,$$

where

$$\varepsilon = \frac{1}{\delta} - \frac{\delta \cdot \ell + 2 - \delta}{\delta \cdot (\delta \cdot \ell + 1)} = \frac{\delta - 1}{\delta \cdot (\delta \cdot \ell + 1)} \leq \frac{1}{\delta \cdot \ell + 1} = \frac{1}{\omega} .$$

By taking $\omega = \lceil \log N \rceil$, the LLL algorithm is applied on a lattice of dimension $\omega = \ell \cdot \delta + 1$ and the bit-size of the elements is bounded by $\log B = \mathcal{O}(\ell \cdot \log N) = \mathcal{O}(\log^2 N)$. The root x_0 can be recovered by applying the previous algorithm in a constant number of intervals. In the sequel, we give the asymptotic complexity of the method and we denote by $\tilde{\mathcal{O}}(\log^k N)$ the complexity $\mathcal{O}(\log^{k+\varepsilon} N)$, for any $\varepsilon > 0$.
Using the L^2 algorithm with complexity $\mathcal{O}(\omega^{5+\varepsilon} \log B + \omega^{4+\varepsilon} \log^2 B)$ the asymptotic complexity of the algorithm is therefore:

$$\tilde{\mathcal{O}}(\log^8 N) .$$

Using the $\tilde{L}^1$ algorithm with complexity $\mathcal{O}(\omega^{5+\varepsilon} \log B + \omega^{1+u+\varepsilon} \log^{1+\varepsilon} B)$ the asymptotic complexity is therefore:

$$\tilde{\mathcal{O}}(\log^7 N) .$$

In the sequel we propose a method that reduces the asymptotic complexity of Coppersmith's method.

3 A variant of Coppersmith's Algorithm with Improved Complexity

The general idea of the improvement consists in applying LLL on a matrix with shorter elements. More precisely, we apply LLL on Coppersmith's matrix where only the most significant bits of all coefficients are considered. To this end, the method is composed by two stages. In a first step, one makes sure that all the coefficients in M are smaller than a particular value by performing modular reductions. In a second step, one truncates coefficients in M, then one applies LLL on this truncated matrix. Finally, one translates the obtained information back into the matrix M. The resulted matrix will be LLL-reduced.

3.1 The New Method

In the sequel, we highlight the two stages of the method. Then we justify our choices and exhibit a proof validating them.

First step: It consists in making sure that all the coefficients in M are less than $N^{\ell+1}$. Thereby, in a given column one should reduce all coefficients with the coefficient on the diagonal. More precisely, this can be done by reducing the elements in turn, starting from the bottom right and ending by the top left of the matrix M, row after row.

Second step: It consists in LLL-reducing the matrix M at low cost by considering a truncated matrix. To this end, one takes the integer quotient of all the matrix coefficients by an integer c that will be determined later. We denote by $M' = \left\lfloor \frac{M}{c} \right\rfloor$ this truncated matrix. Thus, one applies LLL on M' and one obtains an LLL-reduced matrix M'^R. We denote by U the unimodular matrix such that $U \cdot M' = M'^R$. Eventually, applying the unimodular matrix U to the matrix M gives an LLL-reduced matrix M^R.

We now prove the consistency of both steps.

For the first step, we consider Coppersmith's bound on X:

$$X < N^{1/\delta - 1/\omega} .$$

As mentioned previously the diagonal elements in M are:

$$d_{ik} = X^{\delta \cdot k + i} \cdot N^{\ell - k} ,$$

for $0 \leq i < \delta$ for all $0 \leq k < \ell$, and $i = 0$ for $k = \ell$.
Therefore, for all such i, k, we have:

$$N^{\ell-1} \leq X^{\delta \cdot \ell} \leq d_{ik} \leq X^{\delta-1} N^{\ell} \leq N^{\ell+1} . \tag{1}$$

As a consequence, all diagonal elements of matrix M lie between $N^{\ell-1}$ and $N^{\ell+1}$. Thus, by reducing all coefficients in M with the coefficient on the diagonal, one makes sure that all coefficients in M are less than $N^{\ell+1}$.

For the second step, one must ensure that taking the integer quotient of all matrix coefficients by c, leaves the matrix invertible. Using (1), one knows that the smallest diagonal coefficient of M is $X^{\delta\cdot\ell} = X^{\omega-1}$. Therefore, by taking

$$c \leq X^{\omega-1}$$

all diagonal elements of M' are lower-bounded by 1, which ensures that $\det(M') \neq 0$. Eventually, one shows the following proposition:

Proposition 1. *Let $M' = \lfloor \frac{M}{c} \rfloor$ be the matrix containing the integer quotient by c of coefficients in matrix M, and U be the unimodular matrix such that $U \cdot M' = M'^R$. By multiplying matrix U with matrix M, the first vector of the resulting matrix $M^R[1]$ has a norm bounded by*

$$2^{(\omega-1)/4} X^{(\omega-1)/2} N^{\delta\cdot\ell\cdot(\ell+1)/(2\delta\ell+2)} \left(1 + \frac{2^{\omega-2}\cdot\omega^2\cdot c}{X^{\omega-1}}\right) .$$

In order to prove this proposition, one first needs to exhibit a bound on the first vector $M'^R[1]$ in the LLL-reduced matrix M'^R, as depicted in the following Lemma.

Lemma 2. *The first vector $M'^R[1]$ obtained by applying LLL on M' is such that:*

$$||M'^R[1]|| \leq \frac{2^{(\omega-1)/4} X^{(\omega-1)/2} N^{\delta\cdot\ell\cdot(\ell+1)/(2\delta\ell+2)}}{c} . \tag{2}$$

Proof. Since elements in matrix M' are the integer quotient by c of elements in matrix M, and since M and M' are triangular matrices, one has the relation

$$\det M' \leq \frac{\det M}{c^{\,\omega}} . \tag{3}$$

Furthermore, the first vector $M'^R[1]$ has a norm bounded by $2^{(\omega-1)/4}(\det M')^{1/\omega}$. Therefore, using inequality (3) one gets the bound:

$$||M'^R[1]|| \leq \frac{2^{(\omega-1)/4}\cdot(\det M)^{1/\omega}}{c} .$$

Moreover, substituting $(\det M)^{1/\omega}$ by $X^{(\omega-1)/2} N^{\delta\cdot\ell\cdot(\ell+1)/(2\delta\ell+2)}$ in previous equation, allows to retrieve bound (2).

□

With the aim of proving Proposition 1 one also needs to have a bound on the matrix norm $||M'^{-1}||$, as depicted in the following Lemma.

Lemma 3. *The matrix norm of M'^{-1} is upper-bounded as follows*

$$||M'^{-1}|| < \frac{\omega\cdot 2^{\omega-2}\cdot c}{X^{\omega-1}} .$$

Proof. see proof in Annex.

One can now prove Proposition 1.

Proof. The matrix M can be decomposed as $M = c \cdot M' + R$ where all elements in R are upper-bounded by c. Therefore one has the relation

$$U \cdot M = c \cdot U \cdot M' + U \cdot R\,.$$

Using the fact that $U \cdot M' = M'^R$, and considering only the first vector $U[1]$ of the matrix U, one gets the following equation

$$U[1] \cdot M = M^R[1] = c \cdot M'^R[1] + U[1] \cdot R\,.$$

Moreover, still using $U \cdot M' = M'^R$, the vector $U[1]$ can be rewritten $U[1] = M'^R[1] \cdot M'^{-1}$, which leads to the equation:

$$M^R[1] = c \cdot M'^R[1] + M'^R[1] \cdot M'^{-1} \cdot R\,. \tag{4}$$

Furthermore, applying Lemma 2 which provides an upper-bound (2) for $||M'^R[1]||$, and using the triangular inequality, one gets the following relation:

$$||M^R[1]|| \leq 2^{(\omega-1)/4} X^{(\omega-1)/2} N^{\delta\cdot\ell\cdot(\ell+1)/(2\delta\ell+2)} \left(1 + \frac{||M'^{-1} \cdot R||}{c}\right)\,.$$

As the norm of matrix R is upper-bounded by $\omega \cdot c$, and again, using the triangular inequality, one obtains

$$||M^R[1]|| \leq 2^{(\omega-1)/4} X^{(\omega-1)/2} N^{\delta\cdot\ell\cdot(\ell+1)/(2\delta\ell+2)} \left(1 + \omega \cdot ||M'^{-1}||\right)\,.$$

Moreover, using Lemma 3, one has the following upper-bound for the value $||(M')^{-1}||$:

$$||M'^{-1}|| < \frac{\omega \cdot 2^{\omega-2} \cdot c}{X^{\omega-1}}\,.$$

Our previous inequality becomes

$$||M^R[1]|| \leq 2^{(\omega-1)/4} X^{(\omega-1)/2} N^{\delta\cdot\ell\cdot(\ell+1)/(2\delta\ell+2)} \left(1 + \frac{2^{\omega-2} \cdot \omega^2 \cdot c}{X^{\omega-1}}\right)$$

which concludes the proof.

□

On one side, the larger the integer c, the smaller the elements in M', and consequently, the less costly the LLL-reduction of M'. One can see on the other side that, the smaller the integer c, the smaller the upper-bound on the norm of the first vector $M^R[1]$. More precisely, one needs to find a first vector that is upper-bounded by $N^\ell/\sqrt{\omega}$ so that the corresponding polynomial $h(x_0)$ holds over the integers. Ideally, one would like to hold a value c so that the corresponding bound on X remains the same as before, *i.e.* $X < N^{1/\delta - 1/\omega}$. In the following, we show that there exists such a value c which enables to find, with reduced complexity, a short vector bounded by $N^\ell/\sqrt{\omega}$ with $X < N^{1/\delta-1/\omega}$. More precisely, we show that taking $c = \frac{X^{\omega-1}}{2^{2\cdot\omega}}$ is a good trade-off.

Proposition 2. *Let M' be such that $M' = \left\lfloor \frac{M}{c} \right\rfloor$ with*

$$c = \frac{X^{\omega-1}}{2^{2\cdot\omega}}\,,$$

and U be the unimodular matrix such that $U \cdot M' = M'^R$. By multiplying matrix U with matrix M, the first vector of the resulting matrix M^R has a norm bounded by $N^\ell/\sqrt{\omega}$, for $X < N^{1/\delta-1/\omega}$.

Proof. By taking $c = \frac{X^{\omega-1}}{2^{2\cdot\omega}}$, and using Proposition 1, one gets the bound

$$||M^R[1]|| \leq 2^{(\omega-1)/4} \cdot X^{(\omega-1)/2} \cdot N^{\delta\cdot\ell\cdot(\ell+1)/(2\delta\ell+2)} \left(1 + \frac{\omega^2}{2^{\omega+2}}\right)\,.$$

From the condition $||M^R[1]|| \leq N^\ell/\sqrt{\omega}$, and since for all $\omega \geq 1$ we have $\frac{\omega^2}{2^{\omega+2}} < 1$, one gets a relation which induces a bound for X:

$$2^{(\omega+3)/4} \cdot X^{(\omega-1)/2} \cdot N^{\delta\cdot\ell\cdot(\ell+1)/(2\delta\ell+2)} \leq \frac{N^\ell}{\sqrt{\omega}}\,.$$

The factors $\sqrt{\omega}$ and $2^{(\omega+3)/4}$ may, as in Section 2, be omitted since they have little effect on subsequent bound. As a consequence, performing the same calculations as in Section 2 allows to retrieve the bound $X < N^{1/\delta-1/\omega}$. Therefore, when considering a solution x_0 bounded by $X < N^{1/\delta-1/\omega}$, the first vector obtained by applying the unimodular matrix U to the matrix M, has a norm bounded by $N^\ell/\sqrt{\omega}$.

□

Since this vector has a norm bounded by $N^\ell/\sqrt{\omega}$, using Lemma 1 the corresponding polynomial equation $h(x)$ is such that $h(x_0) = 0$ over integers. As a consequence, one can solve it and retrieve the solution x_0. Thus, we have the following Corollary:

Corollary 1. *Let $p(x)$ be a polynomial of degree δ in one variable modulo an integer N of unknown factorization. Let X be such that $X \leq N^{1/\delta}$. By performing the approach described in Section 3.1, one can find all integers x_0 with $p(x_0) = 0 \bmod N$ and $|x_0| \leq X$, in time polynomial in $(\log N, 2^\delta)$.*

Eventually, the overall method is depicted in Algorithm 1.

In the sequel, we show that following this approach allows to decrease the asymptotical complexity of the method by a non-negligible factor.

3.2 The Asymptotic Complexity

Accordingly to our method, it is sufficient to apply the LLL algorithm on a lattice of dimension $\omega = \ell \cdot \delta + 1$ and with elements bounded by

$$\frac{N^{\ell+1}}{c} = \frac{N^{\ell+1}}{X^{\omega-1}} < \frac{N^{\ell+1}}{N^{\ell-1}} < N^2$$

instead of $N^{\ell+1}$. Therefore, the elements have a bit-size bounded by $\log B = \mathcal{O}(\log N)$ instead of $\mathcal{O}(\log^2 N)$. Using the L^2 algorithm with complexity $\mathcal{O}(\omega^{5+\varepsilon} \log B + \omega^{4+\varepsilon} \log^2 B)$ the asymptotic

Algorithm 1 Solving $p(x) \equiv 0 \mod N$

Input: Polynomial equation $p(x) \equiv 0 \mod N$, upper-bound X on x_0, dimension ω, divisor $c = \frac{X^{\omega-1}}{2^{2\cdot\omega}}$

```
Construct Coppersmith matrix $M$ of dimension $\omega$ associated to $p(x)$
Perform modular reductions on $M$ using diagonal elements
$M' := \lfloor \frac{M}{c} \rfloor$
$M'^R, U' := LLL(M')$
$M^R := U' \cdot M$
First polynomial in $M^R$ provides the solution $x_0$
return $x_0$
```

complexity of the algorithm is therefore $\tilde{\mathcal{O}}(\log^6 N)$. Using the $\tilde{L}^1$ algorithm with complexity $\mathcal{O}(\omega^{5+\varepsilon} \log B + \omega^{1+u+\varepsilon} \log^{1+\varepsilon} B)$ with $u = 2.376$, the asymptotic complexity is again $\tilde{\mathcal{O}}(\log^6 N)$. We notice that both asymptotic complexities are similar. This is due to the fact that the $\tilde{L}^1$ algorithm was designed in such a way that it is quasi-linear in the bit-length $\log B$ of the entries.

It leads to the following theorem:

Theorem 3. *The asymptotic complexity of Algorithm 1 for finding small roots of* $p(x) \equiv 0 \mod N$ *using* L^2 *or* $\tilde{L}^1$ *algorithms is* $\tilde{\mathcal{O}}(\log^6 N)$.

We recall that complexities with the original method were $\tilde{\mathcal{O}}(\log^8 N)$ using the L^2 algorithm and $\tilde{\mathcal{O}}(\log^7 N)$ using the $\tilde{L}^1$ algorithm.

In the sequel, we devise a method that allows to speed up the exhaustive search which is usually performed to reach Coppersmith's theoretical bound.

4 Efficient Exhaustive Search

According to Coppersmith's Theorem (see Section 2), one can retrieve the solution x_0 if $x_0 < X = N^{1/\delta - 1/\omega}$. When the solution is close to $N^{1/\delta}$, it is well known that in practice the bound $X < N^{1/\delta}$ should not be reached by taking a very large ω, i.e. by using a very large dimension. Indeed, it is better to use a lattice of reasonable dimension and to perform exhaustive search on most significant bits of x until finding the solution x_0. This is done by writing

$$x_0 = 2^k \cdot \alpha_0 + x'_0$$

and considering α polynomials $p_\alpha(x')$, where α is such that $0 \leqslant \alpha \leqslant \frac{X}{2^k}$, and we have:

$$p_\alpha(x') = p(2^k \cdot \alpha + x') .$$

For the case $\alpha = \alpha_0$, the solution x'_0 satisfies $x'_0 < 2^k = X' = N^{1/\delta - 1/\omega}$ and it has a correct size for LLL to find it using a lattice of dimension ω (where ω is relatively small). For each polynomial p_α, one performs LLL on the corresponding Coppersmith's matrix (see Section 2). The solution x_0 is then found for the right value α_0, i.e. for the right polynomial p_{α_0}.
In Section 4.1, we describe a method that allows to take advantage of the LLL performed for the case $\alpha = i$ to reduce complexity of the LLL performed for the case $\alpha = i + 1$. More precisely,

one considers matrices which are equivalent to Coppersmith's matrices. One uses the fact that for different instances $\alpha = 0, \dots, i$ those matrices can be linked one to another by the relation $M_i = M_{i-1} \cdot P = \cdots = M_0 \cdot P^i$ where P is a well-known structured matrix. Our method consists in LLL-reducing M_0, which gives M_0^R. Then, instead of LLL-reducing $M_1 = M_0 \cdot P$, we apply LLL on $M_0^R \cdot P$. We expect this matrix to be almost reduced already since it is the product of an LLL-reduced matrix M_0^R with a matrix P containing small coefficients. This gives matrix M_1^R. Next step consists in applying LLL on $M_1^R \cdot P$ instead of $M_2 = M_0 \cdot P^2$. Thus, we perform this process incrementally, until the solution x_0' is found, *i.e.* until $\alpha = \alpha_0$.
Thereafter, in Section 4.2 we combine this improvement with the truncation approach described in Section 3.

4.1 Exploiting Relations Between Matrices

We start by highlighting the following proposition that gives a relation between two matrix instances for the case $\alpha = i$ and $\alpha = i + 1$.

Proposition 3. *We denote by M_i the matrix used to solve the polynomial $p_i(x') = p(2^k \cdot i + x')$ for the case $\alpha = i$. Let P denote the lower triangular Pascal matrix defined as follows:*

$$P_{s,t} = \binom{s}{t} \quad s, t \ = \ 0, ..., \omega - 1$$

The matrix

$$M_{i+1} \ = \ M_i \cdot P$$

is a suitable matrix to solve the polynomial $p_{i+1}(x')$ for the case $\alpha = i + 1$.

Proof. In Coppersmith's method, the matrix M_i represents the following ω polynomials :

$$M_i \ \cdot \ \left(1 \, , \, \frac{x'}{X'} \, , \, \left(\frac{x'}{X'}\right)^2 \, , \, \dots \, , \, \left(\frac{x'}{X'}\right)^{\omega-1} \right)^T .$$

All those polynomials would admit x_0 as a solution modulo N^ℓ if $\alpha = \alpha_0 = i$.
For the case $\alpha = i + 1$, one tries to solve the polynomial

$$p_{i+1}(x') = p(2^k \cdot (i+1) + x') = p(2^k \cdot i + x' + 2^k) = p_i(x' + 2^k) = p_i(x' + X') \, .$$

Therefore, for the case $\alpha = i+1$, the polynomials are the same as for the case $\alpha = i$, but written in a different basis, *i.e.* the basis $(1, \frac{x'+X'}{X'}, (\frac{x'+X'}{X'})^2, \dots, (\frac{x'+X'}{X'})^{\omega-1})$. Thus, the polynomials are the following:

$$M_i \ \cdot \ \left(1 \, , \frac{x' + X'}{X'} \, , \, \left(\frac{x' + X'}{X'}\right)^2 \, , \, \dots \, , \, \left(\frac{x' + X'}{X'}\right)^{\omega-1} \right)^T . \tag{5}$$

Yet, we need to return to the original representation of the polynomials, *i.e.* in the basis $(1, \frac{x'}{X'}, (\frac{x'}{X'})^2, \ldots, (\frac{x'}{X'})^{\omega-1})$. To this end, we use the following property regarding the lower triangular Pascal matrix P:

$$\begin{pmatrix} 1 \\ \frac{x'+X'}{X'} \\ \left(\frac{x'+X'}{X'}\right)^2 \\ \vdots \\ \left(\frac{x'+X'}{X'}\right)^{\omega-1} \end{pmatrix} = P \cdot \begin{pmatrix} 1 \\ \frac{x'}{X'} \\ \left(\frac{x'}{X'}\right)^2 \\ \vdots \\ \left(\frac{x'}{X'}\right)^{\omega-1} \end{pmatrix} . \tag{6}$$

Therefore, combining (5) and (6) allows to get the following equivalence:

$$M_i \ \cdot \ \begin{pmatrix} 1 \\ \frac{x'+X'}{X'} \\ \left(\frac{x'+X'}{X'}\right)^2 \\ \vdots \\ \left(\frac{x'+X'}{X'}\right)^{\omega-1} \end{pmatrix} = M_i \ \cdot \ P \ \cdot \ \begin{pmatrix} 1 \\ \frac{x'}{X'} \\ \left(\frac{x'}{X'}\right)^2 \\ \vdots \\ \left(\frac{x'}{X'}\right)^{\omega-1} \end{pmatrix} .$$

As a consequence, the matrix $M_{i+1} \ = \ M_i \cdot P$ is a suitable matrix to solve the polynomial $p_{i+1}(x')$ since it contains ω polynomials that would admit x_0 as a solution modulo N^ℓ if $\alpha = \alpha_0 = i + 1$.

□

Remark: The matrix M_{i+1} considered here is not identical to the matrix that would have been used in Coppersmith's method. However, one can show that both matrices are linked by row linear combinations.

Now that we hold a relation between matrices M_i and M_{i+1}, we enlighten the core idea of the method in the following proposition:

Proposition 4. *The multiplication of the reduced matrix M_i^R by the Pascal matrix P*

$$\tilde{M}_{i+1} \ = \ M_i^R \cdot P$$

gives a matrix $\tilde{M}_{i+1}$ to test the case $\alpha = i + 1$. This matrix contains vectors whose norms are close to vector norms of the LLL-reduced matrix M_i^R. Namely, for all $1 \leqslant j \leqslant \omega$ we have:

$$||\tilde{M}_{i+1}[j]|| \ < \ \sqrt{\omega} \cdot 2^\omega \cdot ||M_i^R[j]|| \, .$$

In particular, for the case $i = \alpha_0$ the first vector of $\tilde{M}_{i+1}$ has a norm bounded by $2^\omega \cdot N^\ell$.

Proof. The application of LLL on matrix M_i provides matrices U_i and M_i^R such that

$$U_i \cdot M_i \ = \ M_i^R \, .$$

Therefore, one has

$$U_i \cdot M_i \cdot P \;=\; M_i^R \cdot P\,.$$

Furthermore, using Proposition 3, one gets the following relation:

$$U_i \cdot M_{i+1} \;=\; M_i^R \cdot P\,. \tag{7}$$

Since matrix U_i is unimodular, the matrix product $U_i \cdot M_{i+1}$ remains a basis of the lattice spawned by the original matrix M_{i+1}. Therefore, using equality 7, one deduces that matrix $\tilde{M}_{i+1} = M_i^R \cdot P$ yields the same lattice as M_{i+1} and can be used to solve the polynomial $p_{i+1}(x')$.
Eventually, since matrix $\tilde{M}_{i+1}$ is the product of M_i^R with a matrix P composed of relatively short elements, the elements in $\tilde{M}_{i+1}$ remain close to those in the reduced matrix M_i^R. Indeed, the largest element in P is $\binom{\omega-1}{\frac{\omega-1}{2}}$ which is less than $2^{\omega-1}$. More precisely, the maximal norm of column vectors in P is equal to:

$$\sqrt{2^{2\cdot(\frac{\omega-1}{2})} + 2^{2\cdot(\frac{\omega-1}{2}+1)} + \cdots + 2^{2\cdot(\omega-1)}} < \sqrt{2^{2\omega-1}} < 2^{\omega}\,.$$

Therefore the norm of each vector of $\tilde{M}_{i+1}$ is at most enlarged by a factor $\sqrt{\omega}\cdot 2^{\omega}$ compared to the norm of the corresponding vector in M_i^R, *i.e.* for all $1 \leqslant j \leqslant \omega$ we have $||\tilde{M}_{i+1}[j]|| \;<\; \sqrt{\omega}\cdot 2^{\omega}\cdot ||M_i^R[j]||$. In particular, for $i = \alpha_0$, since the first vector of M_i^R has a norm bounded by $N^{\ell}/\sqrt{\omega}$, the norm of the first vector of $M_i^R \cdot P$ is bounded by $2^{\omega}\cdot N^{\ell}$ which is relatively close to $N^{\ell}/\sqrt{\omega}$.

□

As a consequence of Proposition 4, one can use matrix $\tilde{M}_{i+1}$ to test the case $\alpha = i+1$. We expect the LLL-reduction of $\tilde{M}_{i+1}$ to be less costly than the one of M_{i+1} since it contains elements relatively close to the ones in the LLL-reduced matrix M_i^R. Thus, one can use this property iteratively to elaborate a new method which consists in LLL-reducing M_0 for the case $\alpha = 0$ (using the method described in Section 3). This gives a reduced matrix M_0^R. Then, one performs a multiplication by P and an LLL-reduction of $\tilde{M}_1 = M_0^R \cdot P$, which gives M_1^R. Then, we multiply M_1^R by P, and perform LLL on $\tilde{M}_2 = M_1^R \cdot P$, which gives M_2^R. We then reiterate this process until the solution x_0' is found, *i.e.* until $\alpha = \alpha_0$. The method is depicted in Algorithm 2.

In Section 3 we depicted a method which consisted in truncating the matrix before applying LLL. In the sequel, we outline a heuristic approach which consists in performing similar truncations during the improved scheme of the exhaustive search described in Section 4.1.

4.2 Combining Coppersmith's Algorithm Variant and Improved Exhaustive Search: A Heuristic Approach

During the exhaustive search described in Section 4.1, we perform the LLL algorithm on the matrix $\tilde{M}_{i+1} = M_i^R \cdot P$ for $0 \leqslant i \leqslant \alpha_0 - 1$. In the sequel we analyze the properties of the matrix $\tilde{M}_{i+1}$. We enlighten that one can truncate elements in matrix $\tilde{M}_{i+1}$ for all i and apply LLL on the truncated matrix $\tilde{M}'_{i+1}$, as performed in Section 3. However, as discussed below,

Algorithm 2 Solving $p(x) \equiv 0 \bmod N$ with exhaustive search on α

Input: Equation $p(x) \equiv 0 \bmod N$, upper-bound X' on x'_0, dimension ω, divisor $c := \frac{X'^{\omega-1}}{2^{2\cdot\omega}}$

```
Construct Coppersmith matrix $M_0$ of dimension $\omega$ associated to $p_0(x') = p(x')$
Perform modular reductions on $M_0$ using diagonal elements
$M'_0 := \lfloor \frac{M_0}{c} \rfloor$
$M'^R_0, U'_0 := LLL(M'_0)$
$M^R_0 := U'_0 \cdot M_0$
$\alpha := 0$
Construct lower triangular Pascal Matrix $P$
while First polynomial in $M^R_\alpha$ does not provide the solution $x'_0$ do
    $\tilde{M}_{\alpha+1} := M^R_\alpha \cdot P$
    $M^R_{\alpha+1} := LLL(\tilde{M}_{\alpha+1})$
    $\alpha := \alpha + 1$
end while
$x_0 := X' \cdot \alpha + x'_0$
return $x_0$
```

this approach is heuristic. Indeed, in order to prove the consistency of truncation method in Section 3, two lemmas (Lemma 2 and 3) were required. In our case, one cannot apply those lemmas straightforwardly since matrix $\tilde{M}_{i+1}$ does not have the same properties as matrix M_0 (considered in Section 3). However, in order to justify our method, we will assume the following assumption that is confirmed in practice.

Assumption 1 *a) We denote by $\tilde{M}'_{i+1}$ the matrix where the elements are the integer quotients of elements in $\tilde{M}_{i+1}$ by c. One has the following bounds on the determinant of $\tilde{M}'_{i+1}$ for all i:*

$$\frac{1}{2^{1/\omega}} \cdot \frac{|\det \tilde{M}_{i+1}|}{c^{\,\omega}} \quad \leq \quad |\det \tilde{M}'_{i+1}| \quad \leq \quad 2^{1/\omega} \cdot \frac{|\det \tilde{M}_{i+1}|}{c^{\,\omega}} \, .$$

b) For all i, the matrix norm of $(\tilde{M}_{i+1})^{-1}$ is upper-bounded as follows:

$$||(\tilde{M}_{i+1})^{-1}|| < \frac{1}{\omega \cdot X'^{\,\omega-\beta}} \, ,$$

where β is a parameter that lies between 0 and 1, as discussed in the sequel.

Discussion on Assumption 1.a: For the first matrix M_0, one can say that $|\det \tilde{M}'_0| \leq |\det \tilde{M}_0|/c^{\,\omega}$ since M_0 is upper triangular (see Lemma 2). However for $i \geq 0$, matrix $\tilde{M}_{i+1}$ is not triangular, and the truncation by c could have a considerable effect on the determinant of $\tilde{M}'_{i+1}$. Yet, we emphasize in Assumption 1.a that the truncation by c affects $\tilde{M}_{i+1}$ and M_0 in a quasi-similar way regarding determinants.

Discussion on Assumption 1.b: In practice, one can see that elements in the LLL-reduced matrix M^R_i are balanced. Based on this fact, one acknowledges that the smallest element in matrix $\tilde{M}_{i+1}$ lies in the last column. Indeed, the product M^R_i by P affects and increases the bit-size of elements in all columns of M^R_i apart from the last one that remains untouched since last column of P is $(0, 0, \dots, 0, 1)^T$. Moreover, using Coppersmith's matrices properties, one knows that elements in last column of M_i^R are all divisible by $X'^{\,\omega-1}$. Therefore, the smallest element in $\tilde{M}_{i+1}$ is larger than $X'^{\,\omega-1}$. As a consequence, one can expect that elements in $(\tilde{M}_{i+1})^{-1}$ are likely to be smaller than a value close to $1/X'^{\,\omega-1}$. In practice, the elements are far smaller than $1/X'^{\,\omega-1}$ and the bound is rather $1/(\omega \cdot X'^{\,\omega-\beta})$ where $0 < \beta \leqslant 1$. In Assumption 1.b, the value β is left unspecified. We will see in Section 5 a suitable instantiation of the parameter β.

Under Assumption 1.a, by taking $c \leq (|\det \tilde{M}_{i+1}|)^{1/\omega}/2^{1/\omega^2}$, we have $|\det \tilde{M}'_{i+1}| > 0$ and the truncated matrix $\tilde{M}'_{i+1}$ remains invertible. As before, one applies LLL on $\tilde{M}'_{i+1}$. We obtain an LLL-reduced matrix M'^R_{i+1}. We denote by U_{i+1} the unimodular matrix such that $U_{i+1} \cdot \tilde{M}'_{i+1} = M'^R_{i+1}$. Thus, one needs to justify the following proposition which enlightens that applying the unimodular matrix U_{i+1} to the matrix $\tilde{M}_{i+1}$ gives an LLL-reduced matrix M^R_{i+1}. In particular, under Assumption 1, for $\alpha = \alpha_0$ one gives a norm upper-bound on the the first vector $M^R_{\alpha_0}[1]$ resulting from the application of the unimodular matrix U_{α_0} to the matrix $\tilde{M}_{\alpha_0}$.

Proposition 5. *Let $\tilde{M}'_{\alpha_0}$ be such that $\tilde{M}'_{\alpha_0} = \left\lfloor \frac{\tilde{M}_{\alpha_0}}{c} \right\rfloor$ and U_{α_0} be the unimodular matrix such that $U_{\alpha_0} \cdot \tilde{M}'_{\alpha_0} = M'^R_{\alpha_0}$. Under Assumption 1, by multiplying matrix U_{α_0} with matrix $\tilde{M}_{\alpha_0}$, the first vector of the resulting matrix $M^R_{\alpha_0}$ has a norm bounded by*

$$2^{\frac{(\omega-1)}{4}+\frac{1}{\omega^2}} X'^{(\omega-1)/2} N^{\delta\cdot\ell\cdot(\ell+1)/(2\delta\ell+2)} \left(1 + \frac{c}{X'^{\,\omega-\beta}}\right) .$$

Proof. If we follow the proof of Proposition 1 until equation (4), one gets the following relation:

$$U_{\alpha_0} \cdot \tilde{M}_{\alpha_0} = c \cdot M'^R_{\alpha_0} + M'^R_{\alpha_0} \cdot (\tilde{M}'_{\alpha_0})^{-1} \cdot R_{\alpha_0} . \tag{8}$$

Since Pascal matrix P has determinant equal to 1, and since the LLL-reduction of a matrix does not change the absolute value of its determinant we have:

$$|\det M^R_{\alpha_0}| = |\det \tilde{M}_{\alpha_0}| = |\det \tilde{M}_{\alpha}| = \det M_0 . \tag{9}$$

Furthermore, using Assumption 1.a and relation (9) one gets:

$$|\det M'^R_{\alpha_0}| \leq \frac{2^{1/\omega} \cdot |\det M^R_{\alpha_0}|}{c^{\,\omega}} \leq \frac{2^{1/\omega} \cdot \det M_0}{c^{\,\omega}} .$$

As a consequence, the first vector $M'^R_{\alpha_0}[1]$ has a norm bounded by:

$$||M'^R_{\alpha_0}[1]|| \leq 2^{(\omega-1)/4} \cdot |\det M'^R_{\alpha_0}|^{1/\omega} \leq 2^{(\omega-1)/4} \cdot \frac{2^{1/\omega^2} \cdot (\det M_0)^{1/\omega}}{c} .$$

Since the determinant of M_0 ie equal to $X'^{(\omega-1)/2} N^{\delta\cdot\ell\cdot(\ell+1)/(2\delta\ell+2)}$, one gets:

$$||M'^R_{\alpha_0}[1]|| \leq \frac{2^{(1/\omega^2)+(\omega-1)/4} \cdot X'^{(\omega-1)/2} \cdot N^{\delta\cdot\ell\cdot(\ell+1)/(2\delta\ell+2)}}{c} .$$

Combining relation (8) with previous inequality, and using the triangular inequality, one has the following:

$$||U_{\alpha_0}[1]\cdot \tilde{M}_{\alpha_0}|| = ||M^R_{\alpha_0}[1]|| \leq 2^{\frac{(\omega-1)}{4}+\frac{1}{\omega^2}} \cdot X'^{(\omega-1)/2} \cdot N^{\delta\cdot\ell\cdot(\ell+1)/(2\delta\ell+2)} \left(1 + \frac{||(\tilde{M'}_{\alpha_0})^{-1}\cdot R_{\alpha_0}||}{c}\right).$$

As the norm of matrix R_{α_0} is upper-bounded by $\omega\cdot c$, and again, using the triangular inequality, one obtains

$$||M^R_{\alpha_0}[1]|| \leq 2^{\frac{(\omega-1)}{4}+\frac{1}{\omega^2}} \cdot X'^{(\omega-1)/2} \cdot N^{\delta\cdot\ell\cdot(\ell+1)/(2\delta\ell+2)} \left(1 + \omega\cdot ||(\tilde{M'}_{\alpha_0})^{-1}||\right).$$

Furthermore, using Assumption 1.b one has

$$||(\tilde{M'}_{\alpha_0})^{-1}|| \leq \frac{c}{\omega\cdot X'^{\,\omega-\beta}}.$$

Finally, one gets

$$||M^R_{\alpha_0}[1]|| \leq 2^{\frac{(\omega-1)}{4}+\frac{1}{\omega^2}} X'^{(\omega-1)/2} N^{\delta\cdot\ell\cdot(\ell+1)/(2\delta\ell+2)} \left(1 + \frac{c}{X'^{\,\omega-\beta}}\right).$$

which concludes the proof.

□

As explained earlier, one has to find a trade-off for the value c which allows to find, with reduced-complexity, a short vector bounded by $N^\ell/\sqrt{\omega}$. Ideally, one would like to hold a value c so that the corresponding bound on X' remains the same as before, *i.e.* $X' < N^{1/\delta-1/\omega}$. In the following, we show that taking $c = X'^{\,\omega-\beta}$ is a good trade-off.

Proposition 6. *Let $\tilde{M'}_{\alpha_0}$ be such that $\tilde{M'}_{\alpha_0} = \left\lfloor \frac{\tilde{M}_{\alpha_0}}{c} \right\rfloor$ with*

$$c = X'^{\,\omega-\beta}, \quad 0 < \beta \leqslant 1$$

and U_{α_0} be the unimodular matrix such that $U_{\alpha_0}\cdot\tilde{M'}_{\alpha_0} = M'^R_{\alpha_0}$. Under Assumption 1, by multiplying matrix U_{α_0} with matrix $\tilde{M}_{\alpha_0}$, the first vector of the resulting matrix $M^R_{\alpha_0}$ has a norm bounded by $N^\ell/\sqrt{\omega}$ for $X' < N^{1/\delta-1/\omega}$.

Proof. By taking $c = X'^{\,\omega-\beta}$ and using Proposition 5, one gets the bound

$$||M^R_{\alpha_0}[1]|| \leq 2^{\frac{(\omega-1)}{4}+\frac{1}{\omega^2}+1} \cdot X'^{(\omega-1)/2} \cdot N^{\delta\cdot\ell\cdot(\ell+1)/(2\delta\ell+2)}.$$

From the condition $||M^R_{\alpha_0}[1]|| \leq N^\ell/\sqrt{\omega}$, one gets a relation which induces a bound for X:

$$2^{\frac{(\omega+3)}{4}+\frac{1}{\omega^2}} \cdot X'^{(\omega-1)/2} \cdot N^{\delta\cdot\ell\cdot(\ell+1)/(2\delta\ell+2)} \leq \frac{N^\ell}{\sqrt{\omega}}.$$

The factors $\sqrt{\omega}$ and $2^{\frac{(\omega+3)}{4}+\frac{1}{\omega^2}}$ may, as in Section 2, be omitted since they have little effect on subsequent bound. As a consequence, performing the same calculations as in Section 2 allows to retrieve the bound $X' < N^{1/\delta-1/\omega}$. Therefore, when considering a solution x_0 bounded by $X' < N^{1/\delta-1/\omega}$, the first vector obtained by applying the unimodular matrix U_{α_0} to the matrix $\tilde{M}_{\alpha_0}$, has a norm bounded by $N^\ell/\sqrt{\omega}$.

□

Since this first vector has a norm bounded by $N^{\ell}/\sqrt{\omega}$, the corresponding polynomial h is such that $h(x_0) = 0$ over integers. Therefore, one can solve it and retrieve the solution x_0.

As a consequence, each LLL-reduction applied to matrix $\tilde{M}_{i+1}$ in **while** loop of Algorithm 2 can be performed on a truncated matrix $\tilde{M}'_{i+1}$, with $c = X'^{\,\omega-\beta}$. It results in Algorithm 3.

Algorithm 3 Solving $p(x)$ with exhaustive search on α and variant of Coppersmith's Algorithm

Input: Equation $p(x) \equiv 0 \bmod N$, upper-bound X' on x'_0, dimension ω, divisors $c_1 := \frac{X'^{\omega-1}}{2^{2\cdot\omega}}$ and $c_2 := X'^{\omega-\beta}$

```
Construct Coppersmith matrix $M_0$ of dimension $\omega$ associated to $p_0(x') = p(x')$
Perform modular reductions on $M_0$ using diagonal elements
$M'_0 := \left\lfloor \frac{M_0}{c_1} \right\rfloor$
$M'^R_0, U'_0 := LLL(M'_0)$
$M^R_0 := U'_0 \cdot M_0$
$\alpha := 0$
Construct lower triangular Pascal Matrix $P$
while First polynomial in $M^R_\alpha$ does not provide the solution $x'_0$ do
    $\tilde{M}_{\alpha+1} := M^R_\alpha \cdot P$
    $\tilde{M}'_{\alpha+1} := \left\lfloor \frac{\tilde{M}_{\alpha+1}}{c_2} \right\rfloor$
    $M'^R_{\alpha+1}, U'_{\alpha+1} := LLL(\tilde{M}'_{\alpha+1})$
    $M^R_{\alpha+1} := U'_{\alpha+1} \cdot \tilde{M}_{\alpha+1}$
    $\alpha := \alpha + 1$
end while
$x_0 := X' \cdot \alpha + x'_0$
return $x_0$
```

5 Experimental results

We have implemented the methods using Howgrave-Graham's variant, on Magma Software V2.19-5 for N being 1024-bit and 2048-bit moduli. We used polynomials of the form $p(x) = x^2 + ax + b \equiv 0 \mod N$ with degree $\delta = 2$. According to Coppersmith's Theorem (see Section 2), one can retrieve the solution x_0 if $x_0 < X = N^{1/2}$. More precisely, Coppersmith's method allows to find the solution x'_0 if $x_0 < X' = N^{1/2-1/\omega}$. We have performed several tests depending on the dimension ω. Results are depicted in Table 1 for the case $\lceil \log_2(N) \rceil = 1024$ and in Table 2 for the case $\lceil \log_2(N) \rceil = 2048$. In both tables, the bit-size of Coppersmith's theoretic upper-bound $X = N^{1/2}$ is given in last column. We have noted the bit-size of the bound X' associated to a dimension ω for which the solution x'_0 is found in practice. We give corresponding timings for different applications:

- Time for LLL execution on original Coppersmith's matrix M_0 (never performed in our method)
- Time for LLL execution on truncated Coppersmith's matrix (applied to reduce M'_0 only)
- Time for LLL execution on truncated quasi LLL-reduced matrix (applied on $\tilde{M'}_i$ for $i = 1, ..., \alpha_0$ during exhaustive search)
- Time for the multiplication with the unimodular matrix ($U_i \cdot \tilde{M}_i$ performed for $i = 1, ..., \alpha_0$ during exhaustive search after each LLL-computation of matrix $\tilde{M'}_i$).

We choose to set the divisor c_2 to the value $X'^{\omega-(1/\delta)}$, *i.e.* to set the parameter β to $1/\delta$. Experimentally, we conjecture that it is a good trade off (see discussion on Table 4).
It is worth noticing that since the value c is not significant in itself, one can truncate matrices at negligible cost by taking $c := 2^{\lfloor \log_2(c) \rfloor}$ and performing shifts of $\lfloor \log_2(c) \rfloor$ bits.

Table 1. Timings (in seconds if not specified) as a function of the dimension for $\lceil \log_2(N) \rceil = 1024$.

	$\log_2(\mathbf{X'})$	492	496	500	503	**504**	505	$\log_2(X) = 512$
	Dimension	29	35	51	71	**77**	87	N/A
Original Method	LLL (M_0)	10.6	35.2	355	2338	**4432**	11426	N/A
	Total Timing (days)	128.6 d.	26.7 d.	16.8 d.	13.9 d.	**13.1 d.**	16.9 d.	N/A
New Method	Truncated LLL (M'_0)	1.6	3.5	18.8	94	**150**	436	N/A
	Truncated Exhaus. LLL ($\tilde{M}'_i$)	0.04	0.12	1.4	9.9	**15.1**	46.5	N/A
	Multiplication Unimodular	0.04	0.08	0.4	1.2	**1.7**	3.6	N/A
	Total Timing (hours)	23.3 h.	3.6 h.	2.1 h.	1.6 h.	**1.2 h.**	1.9 h.	N/A

Table 2. Timings (in seconds if not specified) as a function of the dimension for $\lceil \log_2(N) \rceil = 2048$.

	$\log_2(\mathbf{X'})$	994	1004	1007	1011	**1012**	1013	$\log_2(X) = 1024$
	Dimension	35	51	63	85	**91**	101	N/A
Original Method	LLL (M_0)	164	1617	5667	39342	**60827**	125498	N/A
	Total Timing (years)	5584 y.	53.8 y.	23.6 y.	10.2 y.	**7.9 y.**	8.2 y.	N/A
New Method	Truncated LLL (M'_0)	9	48	146	825	**1200**	2596	N/A
	Truncated Exhaus. LLL ($\tilde{M}'_i$)	0.15	1.6	6.2	33	**48**	104	N/A
	Multiplication Unimodular	0.12	0.6	1.5	5.4	**6.5**	11.5	N/A
	Total Timing (days)	3355 d.	26.7 d.	11.7 d.	3.7 d.	**2.6 d.**	2.8 d.	N/A

As depicted in Tables 1 and 2, by increasing the dimension, one can retrieve solutions x_0 that get ever closer to $X = N^{1/2}$. However, beyond a certain point, it is not profitable to increase the dimension since an exhaustive search would end up faster. In our case, the best dimension to use is depicted in bold on both tables. Indeed, one can see that using a larger dimension allows to find a solution which is one bit longer only, for LLL-executions that take more than twice as much time. As a consequence, for $\lceil \log_2(N) \rceil = 1024$, the best trade-off is to use lattices of dimension 77, and perform an exhaustive search on $512 - 504 = 8$ bits. The exhaustive search then takes $150 + (2^8 - 1)(15.1 + 1.7) \approx 1.2$ hours, which is about 262 times faster than the original method which takes $2^8 \times 4432 \approx 13.1$ days. More generally, performing a single LLL-execution takes 150 seconds when truncating the matrix, compared to 4332 seconds using the original method. In the same way, for $\lceil \log_2(N) \rceil = 2048$, the best trade-off is to use lattices of dimension 91, and perform an exhaustive search on $1024-1012 = 12$ bits. The exhaustive search then takes $1200 + (2^{12} - 1)(48 + 6.5) \approx 2.6$ days, which is about 1109 times faster than the original method which takes $2^{12} \times 60827 \approx 7.9$ years. More generally, performing a single LLL-execution takes 1200 seconds when truncating the matrix, compared to 60827 seconds using the original method. As depicted in Table 3, the larger the modulus N, the more significant the speed-up.

Table 3. Global exhaustive search timing using original/new methods for $\lceil \log_2(N) \rceil = 1024$ and 2048.

	$\lceil \log_2(N) \rceil = 512$	$\lceil \log_2(N) \rceil = 1024$	$\lceil \log_2(N) \rceil = 1536$	$\lceil \log_2(N) \rceil = 2048$
Original method	47 minutes	13.1 days	108.5 days	7.9 years
New method	52 seconds	1.2 hours	5.2 hours	2.6 days
Speed up	54	262	502	1109

Furthermore, we emphasize that Assumption 1 is confirmed in 100% of the cases in practice. We give in Table 4, bounds afforded in Assumption 1 and bounds obtained in practice, for the case $\lceil \log_2(N) \rceil = 1024$ and $\omega = 77$. One can see that practical bounds are always contained within hypothetical bounds. More precisely, for Assumption 1.a, there is a factor 2^{22} between practical and hypothetical bounds. The factor is more considerable for Assumption 1.b, where it is up to 2^{153}. This induces that one could use a larger value for c_2 (*i.e.* a smaller value for β). However, experimental tests yield that the benefit would remain small.

Table 4. Bounds obtained in Assumption 1 and in practice (100% of cases) for $\lceil \log_2(N) \rceil = 1024$ and $\omega = 77$.

	Assumption	Practice
Assumption 1.a: $c^{\omega} \cdot \frac{\lvert \det \tilde{M}'_{i+1} \rvert}{\lvert \det \tilde{M}_{i+1} \rvert}$	$\in [2^{\frac{-1}{\omega}}, 2^{\frac{-1}{\omega}}]$	$\in [2^{\frac{-1}{99}}, 2^{\frac{-1}{99}}]$
Assumption 1.b: $\lvert\lvert (\tilde{M}_{i+1})^{-1} \rvert\rvert \cdot X'^{\omega-(1/\delta)}$	$\leqslant \frac{1}{\omega}$	$\leqslant \frac{1}{2^{160}}$

Eventually, in order to visualize what happens in practice, we have depicted in Figures 1 and 2 the norm of each row vectors in different matrices for the case $\lceil \log_2(N) \rceil = 1024$ with dimension $\omega = 77$. In Figure 1, we give the norm of each row vector in the original Coppersmith's matrix M_0 and in the reduced matrix M_0^R. Namely, the norm of vectors in M_0 (resp. in M_0^R) lies between 2^{38000} and 2^{58250} (resp. between 2^{38853} and 2^{38855}). We recall that matrix M_0 is never reduced in our method. In Figure 2, we give the norm of each row vector in four different matrices that are processed in our method: the first truncated matrix M'_0, the reduced matrix M'^R_0, the next truncated matrices $\tilde{M'}_i$ and the reduced matrices M'^R_i. One can see that elements of all four matrices are far smaller than elements in original matrices (Figure 1). The vectors norm lies around 2^{1250} in M'_0 (resp. around 2^{700} in M'^R_0) and it ranges up to 2^{370} in $\tilde{M'}_i$ (resp. to 2^{300} in M'^R_i). It is worth noting that elements in matrix $\tilde{M'}_i$ are smaller than the ones in M'^R_0 because the divisor c is taken larger for the case $\alpha > 0$ (*i.e.* we take $c_2 > c_1$). It is also interesting to notice that elements in $\tilde{M'}_i$ are close to elements in M'^R_i.

This allows to illustrate part of Proposition 4 which yields that matrix $\tilde{M'}_i$ is almost reduced already.

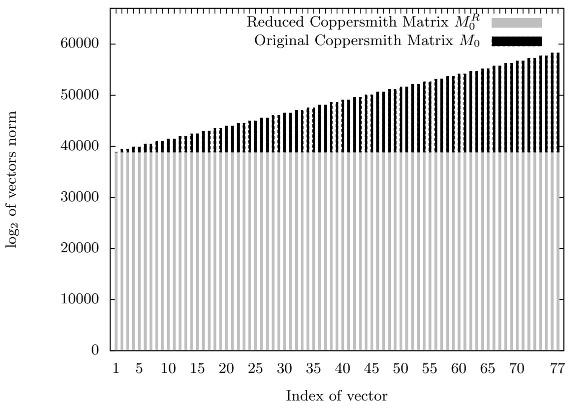

Fig. 1. Norm of vectors in Original Method

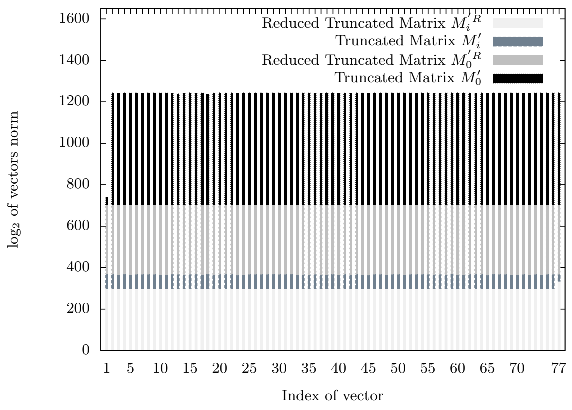

Fig. 2. Norm of vectors in New Method

6 Conclusion

This article brings two improvements that allow to reduce time to find small solutions to polynomial equations. In a first stage, we show that it is not necessary to apply LLL on the original Coppersmith's matrix, but considering a matrix where elements are truncated is sufficient. This allows to divide the asymptotic complexity of the method by a factor $\mathcal{O}(\log^2 N)$ using the L^2 algorithm. In a second phase, we exhibit a new approach to carry out the exhaustive search. It enables to considerably speed up its processing. Experimental results yield that some computations that were completely prohibitive with the original approach are made achievable.

References

1. D. Coppersmith. Finding a small root of a univariate modular equation. In Maurer [11], pages 155–165.
2. Howgrave-Graham. Finding small roots of univariate modular equations revisited. *In Cryptography and Coding*, 1355/1997:131–142, 1997.
3. A.K. Lenstra, H.W. Lenstra, and L. Lovasz. Factoring Polynomials with rational coefficients. *Math. Ann.*, pages 515–534, 1982.
4. P. Nguyen and D. Stehlé. Floating-point LLL revisited. In R. Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 215–233. Springer, 2005.
5. P. Novocin, D. Stehlé, and G. Villard. An LLL-Reduction Algorithm with Quasi-linear Time Complexity. In *Symposium on Theory of Computing*, 2011.
6. D. Coppersmith. Finding a small root of a bivariate integer equation; factoring with high bits known. In Maurer [11], pages 178–189.
7. D. Boneh, G. Durfee, and N.A. Howgrave-Graham. Factoring $N = p^r q$ for large r. In M.J. Wiener, editor, *Advances in Cryptology – CRYPTO '99*, volume 1666 of *LNCS*. Springer, 1999.
8. É. Brier, C. Clavier, J.-S. Coron, and D. Naccache. Cryptanalysis of RSA Signatures with Fixed-Pattern Padding. In J. Kilian, editor, *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *LNCS*, pages 433–439. Springer, 2001.

9. D. Bleichenbacher and A. May. New attacks on RSA with Small Secret CRT-Exponents. In Moti Yung, Yevgeniy Dodis, Aggelos Kiayias, and Tal Malkin, editors, *Public Key Cryptography – PKC 2006*, volume 3958 of *LNCS*. Springer, 2006.
10. D. Boneh and G. Durfee. Cryptanalysis of RSA with private key d less than $N^0.292$. In J. Stern, editor, *Advances in Cryptology – EUROCRYPT '99*, volume 1592 of *LNCS*. Springer, 1999.
11. U. Maurer, editor. *Advances in Cryptology – EUROCRYPT '96*, volume 1070 of *LNCS*. Springer, 1996.

A Proof of Lemma 3

Proof. We denote by $M'_{i,j}$ (resp. $M'^{-1}_{i,j}$) the element in M' (resp. M'^{-1}) situated at the i-th row and j-th column.
We start by showing that, for all i such that $2 \leq i \leq \omega$, we have the inequality

$$-M'^{-1}_{i,i} < M'^{-1}_{i,i-1} \leq 0\,. \tag{10}$$

Indeed, from the relation $M'^{-1} \cdot M' = Id$ and since both matrices are lower triangular, one can write the following equation:

$$M'^{-1}_{i,i-1} \cdot M'_{i-1,i-1} + M'^{-1}_{i,i} \cdot M'_{i,i-1} = 0\,.$$

Therefore one has

$$M'^{-1}_{i,i-1} = -\frac{M'^{-1}_{i,i} \cdot M'_{i,i-1}}{M'_{i-1,i-1}}\,. \tag{11}$$

Moreover, since the first step of the method consisted of successive modular reductions with diagonal elements of M, in a given column, all elements are smaller than the diagonal element. Besides, the same is true for M'. Therefore, for all $j > i$ one has the relation $0 \leq M'_{j,i} < M'_{i,i}$, or equivalently,

$$0 \leq \frac{M'_{j,i}}{M'_{i,i}} < 1\,. \tag{12}$$

Therefore, combining (11) and (12), one gets the relation $-M'^{-1}_{i,i} < -\frac{M'^{-1}_{i,i} \cdot M'_{i,i-1}}{M'_{i-1,i-1}} \leq 0$ which leads to (10).

We now show by recurrence on j that, for all i such that $3 \leq i \leq \omega$ and for all j such that $j \leq i-2$, one has the following relation:

$$-2^{i-j-2} \cdot M'^{-1}_{i,i} < M'^{-1}_{i,j} < 2^{i-j-2} \cdot M'^{-1}_{i,i}\,. \tag{13}$$

Base case: For $j = i-2$, one would like to show that

$$-M'^{-1}_{i,i} < M'^{-1}_{i,i-2} < M'^{-1}_{i,i}\,. \tag{14}$$

From the relation $M'^{-1} \cdot M' = Id$ and since both matrices are lower triangular, one can write the following equation:

$$M'^{-1}_{i,i-2} \cdot M'_{i-2,i-2} + M'^{-1}_{i,i-1} \cdot M'_{i-1,i-2} + M'^{-1}_{i,i} \cdot M'_{i,i-2} = 0\,.$$

Therefore one has the relation

$$M'^{-1}_{i,i-2} = -\frac{M'^{-1}_{i,i-1} \cdot M'_{i-1,i-2}}{M'_{i-2,i-2}} - \frac{M'^{-1}_{i,i} \cdot M'_{i,i-2}}{M'_{i-2,i-2}} \,.$$

Moreover, using (10) and (12), one gets

$$0 \leq -\frac{M'^{-1}_{i,i-1} \cdot M'_{i-1,i-2}}{M'_{i-2,i-2}} < M'^{-1}_{i,i} \,,$$

and from (12) one gets

$$-M'^{-1}_{i,i} < -\frac{M'^{-1}_{i,i} \cdot M'_{i,i-2}}{M'_{i-2,i-2}} \leq 0 \,.$$

Therefore, cumulating these two inequalities, one retrieves relation (14).

Inductive step: We assume that $j \leq i-2$ and that the following relation is true for all k such that $j+1 \leq k \leq i-2$

$$-2^{i-k-2} \cdot M'^{-1}_{i,i} < M'^{-1}_{i,k} < 2^{i-k-2} \cdot M'^{-1}_{i,i} \,. \tag{15}$$

We show that (13) is confirmed when $k = j$.

Using the fact that $M'^{-1} \cdot M' = Id$, one can write the following equation:

$$M'^{-1}_{i,j} \cdot M'_{j,j} + M'^{-1}_{i,j+1} \cdot M'_{j+1,j} + M'^{-1}_{i,j+2} \cdot M'_{j+2,j} + \ \ldots \ + M'^{-1}_{i,i-2} \cdot M'_{i-2,j} + M'^{-1}_{i,i-1} \cdot M'_{i-1,j} + M'^{-1}_{i,i} \cdot M'_{i,j} = 0$$

Therefore one has

$$M'^{-1}_{i,j} = \frac{M'^{-1}_{i,j+1} \cdot M'_{j+1,j}}{M'_{j,j}} + \frac{M'^{-1}_{i,j+2} \cdot M'_{j+2,j}}{M'_{j,j}} + \ \ldots \ + \frac{M'^{-1}_{i,i-2} \cdot M'_{i-2,j}}{M'_{j,j}} + \frac{M'^{-1}_{i,i-1} \cdot M'_{i-1,j}}{M'_{j,j}} + \frac{M'^{-1}_{i,i} \cdot M'_{i,j}}{M'_{j,j}} \tag{16}$$

Following the same argumentation as for the base case, one deduces a relation for the last two elements in (16):

$$-M'^{-1}_{i,i} < \frac{M'^{-1}_{i,i-1} \cdot M'_{i-1,j}}{M'_{j,j}} \ + \ \frac{M'^{-1}_{i,i} \cdot M'_{i,j}}{M'_{j,j}} < M'^{-1}_{i,i} \tag{17}$$

Moreover, the assumption (15) is separately applicable to all the other elements in (16), which gives:

$$-(\sum_{u=0}^{u=i-j-3} 2^u\ ) \cdot M'^{-1}_{i,i} < \frac{M'^{-1}_{i,j+1} \cdot M'_{j+1,j}}{M'_{j,j}} \ + \ \ldots \ + \ \frac{M'^{-1}_{i,i-2} \cdot M'_{i-2,j}}{M'_{j,j}} < (\sum_{u=0}^{u=i-j-3} 2^u\ ) \cdot M'^{-1}_{i,i} \tag{18}$$

Eventually, from (17) and (18), one gets the relation (13), which concludes the inductive proof of relation (13).

Since the value j that maximize bounds in (13) is $j = 0$, one has the relation

$$-2^{i-2} \cdot M'^{-1}_{i,i} < M'^{-1}_{i,j} < 2^{i-2} \cdot M'^{-1}_{i,i} \,. \tag{19}$$

Furthermore, since M' is triangular, one gets for all $i \leq \omega$

$$M'^{-1}_{i,i} = \frac{1}{M'_{i,i}} .$$

Besides, the smallest diagonal element in M' is $M'_{\omega,\omega} = \frac{X^{\delta \cdot \ell}}{c}$. Consequently the largest element in M'^{-1} is met for $i = \omega$ and is equal to

$$M'^{-1}_{\omega,\omega} = \frac{c}{X^{\delta \cdot \ell}} . \tag{20}$$

Therefore, using (19) and (20) one gets the following relation

$$-\frac{2^{\omega-2} \cdot c}{X^{\delta \cdot \ell}} < M'^{-1}_{i,j} < \frac{2^{\omega-2} \cdot c}{X^{\delta \cdot \ell}} . \tag{21}$$

Eventually, one gets the final bound

$$||M'^{-1}|| < \frac{\omega \cdot 2^{\omega-2} \cdot c}{X^{\delta \cdot \ell}} ,$$

which concludes the proof.

□