Finding Small Roots of Bivariate Integer Polynomial Equations: a Direct Approach

Jean-Sébastien Coron

University of Luxembourg

Abstract. Coppersmith described at Eurocrypt 96 an algorithm for finding small roots of bivariate integer polynomial equations, based on lattice reduction. A simpler algorithm was later proposed in [9], but it was asymptotically less efficient than Coppersmith's algorithm. In this paper, we describe an analogous simplification but with the same asymptotic complexity as Coppersmith. We illustrate our new algorithm with the problem of factoring RSA moduli with high-order bits known; in practical experiments our method is several orders of magnitude faster than [9].

Key-words: Coppersmith's theorem, lattice reduction, cryptanalysis.

1 Introduction

At Eurocrypt 96, Coppersmith described how lattice reduction can be used to find small roots of polynomial equations [5–7]. Coppersmith's technique has found numerous applications for breaking variants of RSA; for example, cryptanalysis of RSA with $d < N^{.29}$ [3], polynomial-time factorization of $N = p^r q$ for large r [4], and cryptanalysis of RSA with small secret CRT-exponents [18, 1]. Coppersmith's technique was also used to obtain an improved security proof for OAEP with small public exponent [23], and to show the deterministic equivalence between recovering the private exponent d and factoring N [10, 19].

There are two main theorems from Coppersmith. The first one concerns finding small roots of $p(x) = 0 \mod N$ when the factorization of N is unknown. Coppersmith proved that any root x_0 with $|x_0| < N^{1/\delta}$ can be found in polynomial time, where $\delta = \deg p$. The technique consists in building a lattice that contains the solutions of the modular polynomial equation; all small solutions are shown to belong to an hyperplane of the lattice; an equation of this hyperplane is obtained by considering the last vector of an LLL-reduced basis; this gives a polynomial $h(x)$ such that $h(x_0) = 0$ over the integers, from which one can recover x_0. The method can be extended to handle multivariate modular polynomial equations, but the extension is heuristic only.

Coppersmith's algorithm was further simplified by Howgrave-Graham in [13]. Howgrave-Graham's approach is more direct and consists in building a lattice of polynomials that are multiples of $p(x)$ and N; then by lattice reduction one computes a polynomial with small coefficients such that $h(x_0) = 0 \mod N^k$; if the coefficient of $h(x)$ are sufficiently small then $h(x_0) = 0$ must hold over $\mathbb{Z}$ as well, which enables to recover x_0. Howgrave-Graham's approach seems easier to analyze, in particular for the heuristic extension to multivariate modular equations, for which there is much more freedom in selecting the polynomial multiples than for the univariate case. Howgrave-Graham's approach was actually used in all subsequent applications of Coppersmith's technique [1, 3, 4, 18–20].

Coppersmith's second theorem concerns finding small roots of bivariate integer polynomial equations $p(x, y) = 0$ over the integers (not modulo N). Coppersmith proved that if $|x_0| < X$ and $|y_0| < Y$ with $XY < W^{2/(3\delta)}$ then such root (x_0, y_0) can be found in polynomial-time, where $W := \max_{ij} |p_{ij}|X^i Y^j$. As for the univariate case, the algorithm consists in building a lattice containing the solutions of the polynomial equation; all small solutions are shown to belong to an hyperplane of the lattice, that is obtained by considering the last vector of an LLL-reduced basis. The equation of the hyperplane gives another polynomial $h(x, y)$ with the same root (x_0, y_0) as $p(x, y)$, which enables to recover (x_0, y_0). There can be improved bounds depending on the shape of the polynomial $p(x, y)$; see [2] for a complete analysis. As for the univariate case, the method extends heuristically to more variables. However, as mentioned in [8], the analysis is more difficult to follow than for the univariate case.

For Coppersmith's second theorem, a simplification was later proposed at Eurocrypt 2004 [9], analogous to Howgrave-Graham's simplification for the univariate case. It consists in generating an arbitrary integer n of appropriate size and constructing a lattice of polynomials that are multiples of $p(x, y)$ and n; then by lattice reduction one computes a polynomial with small coefficients such that $h(x_0, y_0) = 0 \mod n$; if the coefficients of $h(x, y)$ are sufficiently small, then $h(x_0, y_0) = 0$ holds over $\mathbb{Z}$, which enables to recover (x_0, y_0) by taking the resultant of $h(x, y)$ and $p(x, y)$. As for the univariate case, this approach seems easier to implement; it was later used in [11] for partial key exposure attacks on RSA, and in [16] to break one variant of RSA.

However, as opposed to the univariate case, this later simplification is not fully satisfactory because asymptotically its complexity is worse than for Coppersmith's second theorem. Namely, the algorithm in [9] is polynomial time under the stronger condition $XY < W^{2/(3\delta)-\varepsilon}$, for any constant $\varepsilon > 0$; but for $XY < W^{2/(3\delta)}$ the algorithm has exponential-time complexity :

$$\exp\left(\mathcal{O}(\log^{2/3} W)\right),$$

whereas Coppersmith's algorithm is polynomial time.

Therefore in this paper we describe a new algorithm for the bivariate integer case, with a simplification analogous to Howgrave-Graham and [9], but with the same polynomial-time complexity as in Coppersmith's algorithm; namely for $XY < W^{2/(3\delta)}$ our algorithm has complexity

$$\mathcal{O}(\log^{15} W)$$

using LLL [17] and $\mathcal{O}(\log^{11} W)$ using the improved L^2 algorithm [21]. This is done by taking a well chosen integer n (rather than arbitrary) when building the lattice of polynomials; this enables to eliminate most columns of the lattice and then apply LLL on a sub-lattice of smaller dimension. Our new algorithm is easy to implement and performs well in practice. In Section 4 we show the results of practical experiments for the factoring with high-order bits known attack against RSA; we show that the running time is improved by several orders of magnitude compared to [9].

2 Preliminaries

Let $u_1, \ldots, u_\omega \in \mathbb{Z}^n$ be linearly independent vectors with $\omega \leq n$. A lattice L spanned by $\langle u_1, \ldots, u_\omega \rangle$ is the set of all integer linear combinations of $u_1, \ldots, u_\omega$. Such a set of vectors u_i's is called a lattice *basis*. We say that the lattice is full rank if $\omega = n$.

Any two bases of the same lattice L are related by some integral matrix of determinant ± 1. Therefore, all the bases have the same Gramian determinant $\det_{1 \leq i,j \leq \omega} < u_i, u_j >$. One defines the *determinant* of the lattice L as the square root of the Gramian determinant. If the lattice L is full rank, then the determinant of L is equal to the absolute value of the determinant of the $\omega \times \omega$ matrix whose rows are the basis vectors $u_1, \ldots, u_\omega$.

Theorem 1 (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:*

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

In order to obtain a better complexity, one can use an improved version of LLL due to Nguyen and Stehlé, called the L^2 algorithm [21]. The L^2 algorithm achieves the same bound on $\|b_1\|$ but in time $\mathcal{O}(\omega^4 n(\omega + \log B) \log B)$.

In this paper we also consider lattices generated by a set of vectors that are not necessarily linearly independent. Let $u_1, \ldots, u_m \in \mathbb{Z}^n$ with $m \geq n$; the lattice L generated by $\langle u_1, \ldots, u_m \rangle$ consists of all integral linear combinations of $u_1, \ldots, u_m$. A lattice basis for L can be obtained by triangularization of $u_1, \ldots, u_m$; a polynomial-time triangularization algorithm is described in [12]; more details will be given in Section 3.1.

We prove a simple lemma that will be useful when analyzing the determinant of such lattices; it shows that the determinant of a full rank lattice generated by a matrix of row vectors is not modified when performing elementary operations on the *columns* of the matrix :

Lemma 1. *Let M be an integer matrix with m rows and n columns, with $m \geq n$. Let L be the lattice generated by the rows of M. Let M' be a matrix obtained by elementary operations on the columns of M, and let L' be the lattice generated by the rows of M'. Then if L is full rank, L' is full rank with $\det L' = \det L$.*

Proof. See Appendix.

3 Our new Algorithm

We consider a polynomial $p(x,y)$ with coefficients in $\mathbb{Z}$ with maximum degree δ independently in x, y :

$$p(x,y) = \sum_{0 \le i,j \le \delta} p_{i,j} x^i y^j.$$

We are looking for an integer pair (x_0, y_0) such that $p(x_0, y_0) = 0$ and $|x_0| < X$ and $|y_0| < Y$. We assume that $p(x,y)$ is irreducible over the integers.

Let k be an integer > 0. We consider the set of polynomials :

$$s_{a,b}(x,y) = x^a \cdot y^b \cdot p(x,y), \text{ for } 0 \le a, b < k \tag{1}$$

$$r_{i,j}(x,y) = x^i \cdot y^j \cdot n, \qquad \text{for } 0 \le i,j < k+\delta \tag{2}$$

where the integer n is generated in the following way.

Let indexes $(i_0,\ j_0)$ be such that $0 \le i_0, j_0 \le \delta$; let S be the matrix of row vectors obtained by taking the coefficients of the polynomials $s_{a,b}(x,y)$ for $0 \le a, b < k$, but only in the monomials $x^{i_0+i} y^{j_0+j}$ for $0 \le i, j < k$. There are k^2 such polynomials $s_{a,b}(x,y)$ and k^2 such monomials, so the matrix S is a square matrix of dimension k^2 (see Figure 1 for an illustration); we take :

$$n := |\det S|.$$

We will show in Lemma 3 that for a well chosen (i_0, j_0), the value $|\det S|$ is lower bounded; in particular, this implies that $|\det S| > 0$ and therefore matrix S is invertible.

$S =$

	x^2y^2	x^2y	xy^2	xy
$s_{1,1}(x,y)$	a	b	c	d
$s_{1,0}(x,y)$		a		c
$s_{0,1}(x,y)$			a	b
$s_{0,0}(x,y)$				a

Fig. 1. Matrix S with $p(x,y) = axy + bx + cy + d$, for $k = 2$ and $(i_0, j_0) = (1,1)$. We get $n = |\det S| = a^4$.

Let $h(x,y)$ be a linear combination of the polynomials $s_{a,b}(x,y)$ and $r_{i,j}(x,y)$. Since we have that $s_{a,b}(x_0,y_0) = 0 \mod n$ for all a, b and $r_{i,j}(x_0,y_0) = 0 \mod n$ for all i, j, we obtain :

$$h(x_0, y_0) = 0 \mod n.$$

The following lemma, due to Howgrave-Graham [13], shows that if the coefficients of polynomial $h(x,y)$ are sufficiently small, then $h(x_0,y_0) = 0$ holds over the integers. For a polynomial $h(x,y) = \sum_{i,j} h_{ij} x^i y^j$, we define $\|h(x,y)\|^2 := \sum_{i,j} |h_{ij}|^2$.

Lemma 2 (Howgrave-Graham). *Let $h(x,y) \in \mathbb{Z}[x,y]$ which is a sum of at most ω monomials. Suppose that $h(x_0,y_0) = 0 \mod n$ where $|x_0| \le X$ and $|y_0| \le Y$ and $\|h(xX,yY)\| < n/\sqrt{\omega}$. Then $h(x_0,y_0) = 0$ holds over the integers.*

Proof. We have:

$$\begin{aligned}
|h(x_0,y_0)| &= \left|\sum h_{ij} x_0^i y_0^i\right| = \left|\sum h_{ij} X^i Y^j \left(\frac{x_0}{X}\right)^i \left(\frac{y_0}{Y}\right)^j\right| \\
&\le \sum \left|h_{ij} X^i Y^j \left(\frac{x_0}{X}\right)^i \left(\frac{y_0}{Y}\right)^j\right| \le \sum \left|h_{ij} X^i Y^j\right| \\
&\le \sqrt{\omega}\|h(xX,yY)\| < n
\end{aligned}$$

Since $h(x_0,y_0) = 0 \mod n$, this gives $h(x_0,y_0) = 0$. □

We consider the lattice L generated by the row vectors formed with the coefficients of polynomials $s_{a,b}(xX,yY)$ and $r_{i,j}(xX,yY)$. In total, there are $k^2 + (k+\delta)^2$ such polynomials; moreover these polynomials are of maximum degree $\delta + k - 1$ in $x,\ y$, so they contain at most $(\delta+k)^2$ coefficients. Let M be the corresponding matrix of row vectors; M is therefore a rectangular matrix with $k^2 + (k+\delta)^2$ rows and $(k+\delta)^2$ columns (see Figure 2 for an illustration). Observe that the rows of M do not form a basis of L (because there are more rows than columns), but L is a full rank lattice of dimension $(k+\delta)^2$ (because the row vectors corresponding to polynomials $r_{i,j}(xX,yY)$ form a full rank lattice).

$$M = \begin{array}{r||cccc|ccccc}
 & x^2y^2 & x^2y & xy^2 & xy & x^2 & y^2 & x & y & 1 \\
\hline
s_{1,1}(xX,yY) & aX^2Y^2 & bX^2Y & cXY^2 & dXY & & & & & \\
s_{1,0}(xX,yY) & & aX^2Y & & cXY & bX^2 & & dX & & \\
s_{0,1}(xX,yY) & & & aXY^2 & bXY & & cY^2 & & dY & \\
s_{0,0}(xX,yY) & & & & aXY & & & bX & cY & d \\
\hline
r_{2,2}(xX,yY) & nX^2Y^2 & & & & & & & & \\
r_{2,1}(xX,yY) & & nX^2Y & & & & & & & \\
r_{1,2}(xX,yY) & & & nXY^2 & & & & & & \\
r_{1,1}(xX,yY) & & & & nXY & & & & & \\
r_{2,0}(xX,yY) & & & & & nX^2 & & & & \\
r_{0,2}(xX,yY) & & & & & & nY^2 & & & \\
r_{1,0}(xX,yY) & & & & & & & nX & & \\
r_{0,1}(xX,yY) & & & & & & & & nY & \\
r_{0,0}(xX,yY) & & & & & & & & & n
\end{array}$$

Fig. 2. Lattice of polynomials with $p(x,y) = axy + bx + cy + d$, for $k = 2$ and $(i_0, j_0) = (1,1)$

Let L_2 be the sublattice of L where the coefficients corresponding to all monomials of the form $x^{i_0+i} y^{j_0+j}$ with $0 \le i,j < k$ are set to zero (those monomials correspond to the matrix left-hand block in Fig. 2). There are k^2

such monomials, so L_2 is a full rank lattice of dimension :

$$\omega = (\delta + k)^2 - k^2 = \delta^2 + 2 \cdot k \cdot \delta. \tag{3}$$

A matrix basis for L_2 can be obtained by first triangularizing M using elementary row operations and then taking the corresponding submatrix (see Fig. 3). A polynomial-time triangularization algorithm is described in [12]; more details will be given in Section 3.1.

	x^2y^2	x^2y	xy^2	xy	x^2	y^2	x	y	1
$s_{1,1}(xX, yY)$	aX^2Y^2	bX^2Y	cXY^2	dXY					
$s_{1,0}(xX, yY)$		aX^2Y		cXY	bX^2	dX			
$s_{0,1}(xX, yY)$			aXY^2	bXY		cY^2		dY	
$s_{0,0}(xX, yY)$				aXY			bX	cY	d
$q_0(xX, yY)$					$*$	$*$	$*$	$*$	$*$
$q_1(xX, yY)$						$*$	$*$	$*$	$*$
$q_2(xX, yY)$							$*$	$*$	$*$
$q_3(xX, yY)$								$*$	$*$
$q_4(xX, yY)$									$*$

Fig. 3. Triangularized lattice of polynomials with $p(x, y) = axy + bx + cy + d$, for $k = 2$ and $(i_0, j_0) = (1, 1)$. The 5 polynomials $q_i(xX, yY)$ generate lattice L_2, with coefficients only in the 5 monomials x^2, y^2, x, y and 1. Algorithm LLL is applied on the corresponding 5-dimensional lattice.

We apply the LLL algorithm on lattice L_2. From theorem 1, we obtain a non-zero polynomial $h(x, y)$ that satisfies $h(x_0, y_0) = 0 \mod n$ and :

$$\|h(xX, yY)\| \le 2^{(\omega-1)/4} \cdot \det(L_2)^{1/\omega}. \tag{4}$$

From lemma 2, this implies that if :

$$2^{(\omega-1)/4} \cdot \det(L_2)^{1/\omega} \le \frac{n}{\sqrt{\omega}}, \tag{5}$$

then $h(x_0, y_0) = 0$ must hold over the integers.

Now we claim that polynomial $h(x, y)$ cannot be a multiple of $p(x, y)$. Assume the contrary; then the row vector coefficients of $h(x, y)$ is a linear combination of the row vector coefficients of polynomials $s_{a,b}(x, y)$ only. Given that matrix S contains the coefficients of $s_{a,b}(x, y)$ for monomials $x^{i+i_0}y^{j+j_0}$ and given that $h(x, y)$ does not contain such monomials (because $h(x, y)$ lies in L_2), this gives a linear combination of the rows of S equal to zero with non-zero coefficients; a contradiction since matrix S is invertible.

The polynomial $p(x, y)$ being irreducible, this implies that $p(x, y)$ and $h(x, y)$ are algebraically independent with a common root (x_0, y_0); therefore, taking :

$$Q(x) = \text{Resultant}_y(h(x, y), p(x, y))$$

gives a non-zero integer polynomial such that $Q(x_0) = 0$. Using any standard root-finding algorithm, we can recover x_0, and finally y_0 by solving $p(x_0, y) = 0$. This terminates the description of our algorithm.

It remains to compute the determinant of lattice L_2. First we consider the same matrices of row vectors as previously, except that we remove the $X^i Y^j$ powers. Therefore let M' be the same matrix as M, except that we take the coefficients of polynomials $s_{a,b}(x, y)$ and $r_{i,j}(x, y)$, instead of $s_{a,b}(xX, yY)$ and $r_{i,j}(xX, yY)$; matrix M' has $k^2 + (k+\delta)^2$ rows and $(k+\delta)^2$ columns. We put the coefficients corresponding to monomials $x^{i+i_0} y^{j+j_0}$ for $0 \le i, j < k$ on the left hand block, which has therefore k^2 columns; matrix M' has then the following form :

$$M' = \begin{bmatrix} S & T \\ nI_{k^2} & 0 \\ 0 & nI_w \end{bmatrix}$$

where S is the previously defined square matrix of dimension k^2, while T is a matrix with k^2 rows and $\omega = k^2 + 2k\delta$ columns. Let L' be the lattice generated by the rows of M', and let L'_2 be the sublattice where all coefficients corresponding to monomials $x^{i+i_0} y^{j+j_0}$ for $0 \le i, j < k$ are set to zero. Note that lattice L' corresponds to lattice L without the $X^i Y^j$ powers, whereas lattice L'_2 corresponds to lattice L_2.

Since $n = |\det S|$, we can find an integer matrix S' satisfying :

$$S' \cdot S = nI_{k^2},$$

namely S' is (up to sign) the adjoint matrix (or comatrix) of S, verifying $S' \cdot S = (\det S) I_{k^2}$. By elementary operations on the rows of M', we can therefore subtract $S' \cdot S$ to the nI_{k^2} block of M'; this gives the following matrix :

$$M'_2 = \begin{bmatrix} I_{k^2} & 0 & 0 \\ -S' & I_{k^2} & 0 \\ 0 & 0 & I_\omega \end{bmatrix} \cdot M' = \begin{bmatrix} S & T \\ 0 & T' \\ 0 & nI_\omega \end{bmatrix}, \tag{6}$$

where $T' = -S' \cdot T$ is a matrix with k^2 rows and ω columns. By elementary operations on the rows of M'_2, we obtain :

$$M'_3 = U \cdot M'_2 = \begin{bmatrix} S & T \\ 0 & T'' \\ & 0 \end{bmatrix},$$

where T'' is a square matrix of dimension ω. We obtain that T'' is a row matrix basis of lattice L'_2, which gives :

$$\det L' = |\det \begin{bmatrix} S & T \\ 0 & T'' \end{bmatrix}| = |\det S| \cdot |\det T''| = |\det S| \cdot \det L'_2 = n \cdot \det L'_2. \tag{7}$$

We now proceed to compute $\det L'$. The polynomial $p(x, y)$ being irreducible, the gcd of its coefficients is equal to 1. This implies that by elementary operation of the *columns* of M', we can obtain a matrix whose left upper $k^2 \times k^2$ block is the identity matrix and the right upper block is zero. From lemma 1, this does not change the determinant of the generated lattice. Let V be the corresponding unimodular transformation matrix of dimension $(\delta+k)^2$; this gives :

$$M'_4 = M' \cdot V = \begin{bmatrix} I_{k^2} & 0 \\ & nV \end{bmatrix}.$$

By elementary row operations on M'_4 based on V^{-1} we obtain :

$$M'_5 = \begin{bmatrix} I_{k^2} & 0 \\ 0 & V^{-1} \end{bmatrix} \cdot M'_4 = \begin{bmatrix} I_{k^2} & 0 \\ & nI_{(\delta+k)^2} \end{bmatrix} = \begin{bmatrix} I_{k^2} & 0 \\ nI_{k^2} & 0 \\ 0 & nI_\omega \end{bmatrix},$$

which again by elementary row operations gives :

$$M'_6 = U' \cdot M'_5 = \begin{bmatrix} I_{k^2} & 0 \\ 0 & nI_\omega \\ 0 & 0 \end{bmatrix}.$$

Finally this implies :

$$\det L' = \det \begin{bmatrix} I_{k^2} & 0 \\ 0 & nI_\omega \end{bmatrix} = n^\omega \tag{8}$$

Combining equations (7) and (8), we obtain :

$$\det L'_2 = n^{\omega-1}.$$

Recall that the columns of L'_2 correspond to monomials $x^i y^j$ for $0 \le i, j < \delta + k$, excluding monomials $x^{i_0+i} y^{j_0+j}$ for $0 \le i, j < k$. The columns of lattice L_2 are obtained from the columns of L'_2 by multiplication with the corresponding $X^i Y^j$ powers; this gives :

$$\begin{aligned} \det L_2 &= \det L'_2 \cdot \frac{\prod\limits_{0 \le i,j < \delta+k} X^i Y^j}{\prod\limits_{0 \le i,j<k} X^{i_0+i} Y^{j_0+j}} \\ &= n^{\omega-1} \cdot \frac{(XY)^{(\delta+k-1)\cdot(\delta+k)^2/2-(k-1)\cdot k^2/2}}{(X^{i_0} Y^{j_0})^{k^2}} \end{aligned}$$

From inequality (5) we obtain the following condition for Howgrave-Graham's lemma to apply :

$$2^{\omega\cdot(\omega-1)/4} \cdot \frac{(XY)^{(\delta+k-1)\cdot(\delta+k)^2/2-(k-1)\cdot k^2/2}}{(X^{i_0} Y^{j_0})^{k^2}} \le \frac{n}{\omega^{\omega/2}}. \tag{9}$$

It remains to bound $n = |\det S|$ as a function of the coefficients of $p(x, y)$. Let

$$W = \max_{i,j} |p_{ij}| X^i Y^j$$

The following lemma shows that for the right choice of (i_0, j_0), the determinant of S is bounded in absolute value :

Lemma 3. *Given (u, v) such that $W = |p_{uv}|X^u Y^v$, let indices (i_0, j_0) that maximize the quantity $8^{(i-u)^2+(j-v)^2}|p_{ij}|X^i Y^j$. Then*

$$\left(\frac{W}{X^{i_0}Y^{j_0}}\right)^{k^2} 2^{-6k^2\delta^2-2k^2} \leq |\det S| \leq \left(\frac{W}{X^{i_0}Y^{j_0}}\right)^{k^2} \cdot 2^{k^2}. \tag{10}$$

Proof. The proof is very similar to the proof of Lemma 3 in [7]; see Appendix B.

Combining inequalities (9) and (10) with $n = |\det S|$ and $\sqrt{\omega} \leq 2^{\omega/2}$, we obtain the sufficient condition :

$$2^{\omega\cdot(\omega-1)/4} \cdot (XY)^{(\delta+k-1)\cdot(\delta+k)^2/2-(k-1)\cdot k^2/2} \leq W^{k^2} \cdot 2^{-6k^2\delta^2-2k^2} \cdot 2^{-\omega^2/2}.$$

This condition is satisfied if :

$$XY < W^{\alpha} \cdot 2^{-9\delta},$$

where

$$\alpha = \frac{2k^2}{\delta \cdot (3k^2 + k(3\delta - 2) + \delta^2 - \delta)}.$$

Finally we obtain the sufficient condition :

$$XY < W^{2/(3\delta)-1/k} \cdot 2^{-9\delta}. \tag{11}$$

The running time is dominated by the time it takes to run LLL on a lattice of dimension $\delta^2 + 2k\delta$, with entries bounded by $\mathcal{O}(W^{k^2})$. Namely, the entries of a matrix basis for L_2 can be reduced modulo $n \cdot X^i Y^j$ on the columns corresponding to monomial $x^i y^j$, because of polynomials $r_{ij}(xX, yY) = n \cdot X^i Y^j x^i y^j$. This implies that we can obtain a matrix basis for L_2 whose entries are bounded by $\mathcal{O}(nX^{\delta+k}Y^{\delta+k})$. From inequality (10) we have $n = \mathcal{O}(W^{k^2})$; using (11) this implies that the matrix entries can be bounded by $\mathcal{O}(W^{k^2})$. From theorem 1 and taking $k > \delta$, the running time is therefore bounded by :

$$\mathcal{O}\left(\delta^6 k^{12} \log^3 W\right)$$

using the LLL algorithm, and $\mathcal{O}\left(\delta^5 k^9 \log^2 W\right)$ using the improved L^2 algorithm.

Finally, under the weaker condition

$$XY < W^{2/(3\delta)},$$

one can set $k = \lfloor \log W \rfloor$ and do exhaustive search on the high order $\mathcal{O}(\delta)$ unknown bits of x_0. The running time is then polynomial in 2^δ and $\log W$. Moreover, for a fixed δ, the running time is $\mathcal{O}(\log^{15} W)$ using the LLL algorithm, and $\mathcal{O}(\log^{11} W)$ using the improved L^2 algorithm. Thus we have shown :

Theorem 2 (Coppersmith). *Let $p(x, y)$ be an irreducible polynomial in two variables over $\mathbb{Z}$, of maximum degree δ in each variable separately. Let X and Y be upper bounds on the desired integer solution (x_0, y_0), and let $W = \max_{i,j} |p_{ij}|X^i Y^j$. If $XY < W^{2/(3\delta)}$, then in time polynomial in $(\log W, 2^\delta)$, one can find all integer pairs (x_0, y_0) such that $p(x_0, y_0) = 0$, $|x_0| \leq X$, and $|y_0| \leq Y$.*

As in [7], there can be improved bounds depending on the shape of the polynomial $p(x,y)$:

Theorem 3 (Coppersmith). *With the hypothesis of Theorem 2, except that $p(x,y)$ has total degree δ, the appropriate bound is :*

$$XY < W^{1/\delta}.$$

Proof. See the full version of this paper.

3.1 Computing a Basis of L_2

In the previous section one needs to compute a basis for lattice L_2, which is then given as input to the LLL algorithm. Such lattice basis can be obtained by triangularization of matrix M; a matrix A is upper triangular if $A_{ij} = 0$ for $i > j$ (as illustrated in Figure 3). A triangularization algorithm is described in [12]; for an $m \times n$ matrix of row vectors, its running time is $\mathcal{O}(n^{3+\varepsilon} m \log^{1+\varepsilon} B)$ for any $\varepsilon > 0$, when the matrix entries are bounded by B in absolute value.

Observe that we don't need to triangularize the full matrix M. Namely from our analysis of the previous section, equation (6) can be used to obtain a set of row vectors that generate L_2; a triangularization algorithm is then applied to derive a lattice basis for L_2. For this we need to compute matrix S' such that $S' \cdot S = (\det S) \cdot I$; we note that this is implemented in Shoup's NTL library [22].

Another possibility is to compute the Hermite Normal form (HNF) of M. An $m \times n$ matrix A of rank n is in HNF if it is upper triangular and $a_{ii} > 0$ for all $1 \leq i \leq n$ and $0 \leq a_{ij} < a_{jj}$ for all $1 \leq j \leq n$ and $1 \leq i < j$. A classical result says that if an $m \times n$ matrix M is of rank n then there exists a $m \times m$ unimodular matrix U such that $U \cdot M$ is in HNF; moreover the HNF is unique. An algorithm for computing the HNF is also described in [12], with the same asymptotic complexity as triangularization. A HNF algorithm is also implemented in Shoup's NTL library [1].

3.2 Difference with the Algorithm in [9]

In [9] a similar lattice L is built but with an integer n which is co-prime with the constant coefficient of $p(x,y)$. This implies that the full lattice L must be considered, whose dimension $d_L = (\delta + k)^2$ grows quadratically with k instead of linearly as in our sub-lattice of dimension $\omega = \delta^2 + 2k\delta$.

With the full lattice L the LLL fudge factor is then $2^{(d_L-1)/4} = 2^{\mathcal{O}(k^2)}$ instead of $2^{(\omega-1)/4} = 2^{\mathcal{O}(k)}$. This translates in the bound for XY into the condition $XY < W^{2/(3\delta)-1/k} \cdot 2^{-\mathcal{O}(k^2+\delta)}$ instead of $XY < W^{2/(3\delta)-1/k} \cdot 2^{-9\delta}$. This implies that in [9], in order to reach the bound $XY < W^{2/(3\delta)}$, one must do exhaustive

[1] The LLL algorithms implemented in Shoup's NTL library can in principle receive as input a matrix with $m \geq n$, but for large dimensions we got better results when a lattice basis was provided instead.

search on the high order $\mathcal{O}((\log W)/k + k^2)$ bits of X. The optimum is to take $k := \mathcal{O}(\log^{1/3} W)$; this gives a sub-exponential time complexity :

$$\exp\left(\mathcal{O}(\log^{2/3} W)\right),$$

instead of the polynomial-time complexity as in Coppersmith's algorithm and our new algorithm.

3.3 Extension to more Variables

Our algorithm can be extended to solve integer polynomial equations with more than two variables, but as for Coppersmith's algorithm, the extension is heuristic only.

Let $p(x, y, z)$ be a polynomial in three variables over the integers, of degree δ independently in x, y and z. Let (x_0, y_0, z_0) be an integer root of $p(x, y, z)$, with $|x_0| \le X$, $|y_0| \le Y$ and $|z_0| \le Z$. As for the bivariate case, we can select indices (i_0, j_0, k_0) that maximize the quantity $X^i Y^j Z^k |p_{ijk}|$ and consider the matrix S formed by the coefficients of polynomials $s_{abc}(x, y, z) = x^a y^b z^c \cdot p(x, y, z)$ for $0 \le a, b, c < m$ for some parameter m, but only in the monomials $x^{i_0+i} y^{j_0+j} z^{k_0+k}$ for $0 \le i, j, k < m$. Then we take $n := |\det S|$ and define the additional polynomials $r_{ijk}(x, y, z) = x^i y^j z^k n$ for $0 \le i, j, k < \delta + m$. Then one builds the lattice L formed by all linear combinations of polynomials $s_{abc}(xX, yY, zZ)$ and $r_{ijk}(xX, yY, zZ)$, and consider the sublattice L_2 obtained by setting to 0 the coefficients of monomials corresponding to matrix S. Lattice L_2 has dimension $\omega = (\delta + m)^3 - m^3$ and using the same analysis as in Section 3, one obtains that $\det L'_2 = n^{\omega-1}$ where L'_2 is the same lattice as L_2 but without the $X^i Y^j Z^k$ powers.

One then applies LLL to sublattice L_2; if the ranges X, Y, Z are small enough, we are guaranteed to find a polynomial $h_1(x, y, z)$ such that $h_1(x_0, y_0, z_0) = 0$ over $\mathbb{Z}$ and $h_1(x, y, z)$ is not a multiple of $p(x, y, z)$, but this is not enough. The second vector produced by LLL gives us a second polynomial $h_2(x, y, z)$ that can satisfy the same property by bounding its norm as in [3]. One can then take the resultant between the three polynomials $p(x, y, z)$, $h_1(x, y, z)$ and $h_2(x, y, z)$ in order to obtain a polynomial $f(x)$ such that $f(x_0) = 0$. But we have no guarantee that the polynomials $h_1(x, y, z)$ and $h_2(x, y, z)$ will be algebraically independent; this makes the method heuristic only.

4 Practical Experiments

As mentioned previously, a direct application of Coppersmith's theorem for the bivariate integer case is to factor $N = pq$ when half of the most significant bits (or least significant bits) of p are known.

Theorem 4 (Coppersmith [7]). *Given $N = pq$ and the high-order $1/4 \log_2 N$ bits of p, one can recover the factorization of N in time polynomial in $\log N$.*

Namely, given the most significant bits of p, one can write :

$$N = (P_0 + x) \cdot (Q_0 + y),$$

where P_0 and Q_0 contain the most significant bits of p and q. This gives a bivariate integer polynomial equation, for which Theorem 2 can be applied directly. One gets $W = P_0 \cdot X \simeq N^{1/2} \cdot X$ which gives $XY < W^{2/3} \simeq N^{1/3} \cdot X^{2/3}$. With $X = Y$ this gives $|x_0| \leq X = N^{1/4}$.

The result of practical experiments are summarized in Table 1, using Shoup's NTL library [22]. For comparison we have implemented our algorithm and the algorithm in [9]. Table 1 shows that our new algorithm is significantly more efficient; for example, for a 1024-bits modulus with $282 = 256 + 26$ bits of p given, our algorithm takes 1 second instead of 13 minutes for the algorithm in [9]; this is due to the fact that LLL is applied on a lattice of smaller dimension.

Parameters			New algorithm		Algorithm in [9]	
N	k	bits of p given	Dimension	LLL	Dimension	LLL
512 bits	4	144 bits	9	<1 s	25	20 s
512 bits	5	141 bits	11	<1 s	36	2 min
1024 bits	5	282 bits	11	1 s	36	13 min
1024 bits	12	266 bits	25	42 s	169	-

Table 1. Running times for factoring $N = pq$ given the high-order bits of p, using our algorithm and the algorithm in [9], with Shoup's NTL library on a 1.6 GHz PC under Linux

The problem of factoring $N = pq$ given the high-order (or low-order) bits of p can also be solved using a simple variant of the one variable modular case, as shown by Howgrave-Graham in [13]. Therefore we have also implemented Howgrave-Graham's algorithm to provide a comparison; experimental results are given in Table 2. We obtain that for the particular case of factoring with high-order bits known, our algorithm and Howgrave-Graham's algorithm have roughly the same running time, and work with the same lattice dimension (but the two lattices are different).

5 Conclusion

We have described a new algorithm for finding small roots of bivariate polynomial equations over the integers, which is simpler than Coppersmith's algorithm but with the same asymptotic complexity. Our simplification is analogous to the simplification brought by Howgrave-Graham for the univariate modular case; it improves on the algorithm in [9] which was not polynomial-time for certain parameters. In practical experiments, our algorithm performs several order of magnitude faster than the algorithm in [9].

N	k	bits of p given	Dimension	LLL
512 bits	4	144 bits	9	<1 s
512 bits	5	141 bits	11	<1 s
1024 bits	5	282 bits	11	1 s
1024 bits	12	266 bits	25	37 s

Table 2. Running times for factoring $N = pq$ given the high-order bits of p, using Howgrave-Graham's algorithm with Shoup's NTL library on a 1.6 GHz PC running under Linux.

References

1. D. Bleichenbacher and A. May, *New Attacks on RSA with Small Secret CRT-Exponents*. In Practice and Theory in Public Key Cryptography (PKC 2006), Lecture Notes in Computer Science, Springer-Verlag, 2006.
2. J. Blomer and Alexander May, *A Tool Kit for Finding Small Roots of Bivariate Polynomials over the Integers*, In Advances in Cryptology (Eurocrypt 2005), Lecture Notes in Computer Science Volume 3494, pages 251-267, Springer-Verlag, 2005.
3. D. Boneh and G. Durfee, *Crypanalysis of RSA with private key d less than $N^{0.292}$*, proceedings of Eurocrypt '99, vol. 1592, Lecture Notes in Computer Science.
4. D. Boneh, G. Durfee and N.A. Howgrave-Graham, *Factoring $n = p^r q$ for large r*, proceedings of Crypto '99, vol. 1666, Lecture Notes in Computer Science.
5. D. Coppersmith, *Finding a Small Root of a Univariate Modular Equation*, proceedings of Eurocrypt '96, vol. 1070, Lecture Notes in Computer Science.
6. D. Coppersmith, *Finding a Small Root of a Bivariate Integer Equation; Factoring with High Bits Known*, proceedings of Eurocrypt' 96, vol. 1070, Lecture Notes in Computer Science.
7. D. Coppersmith, *Small solutions to polynomial equations, and low exponent vulnerabilities*. J. of Cryptology, 10(4)233-260, 1997. Revised version of two articles of Eurocrypt '96.
8. D. Coppersmith, *Finding small solutions to small degree polynomials*. In Proc. of CALC '01, LNCS, Sptinger-Verlag, 2001.
9. J.S. Coron, *Finding Small Roots of Bivariate Polynomial Equations Revisited*. Proceedings of Eurocrypt 2004, LNCS, Springer-Verlag, 2004.
10. J.S. Coron and A. May, *Deterministic Polynomial-Time Equivalence of Computing the RSA Secret Key and Factoring*, Journal of Cryptology, Volume 20, Number 1, January 2007.
11. M. Ernst, E. Jochemsz, A. May and B. de Weger, *Partial Key Exposure Attacks on RSA up to Full Size Exponents*. In Advances in Cryptology (Eurocrypt 2005), Lecture Notes in Computer Science Volume 3494, pages 371-386, Springer-Verlag, 2005.
12. J. Hafner and K. McCurley, *Asymptotically fast triangularization of matrices over rings*, SIAM J. Comput. 20 (1991), 1068-1083.
13. N. A. Howgrave-Graham, *Finding small roots of univariate modular equations revisited*. In Cryptography and Coding, volume 1355 of LNCS, pp. 131-142. Springer Verlag, 1997.
14. N. A. Howgrave-Graham, *Approximate integer common divisors*. In Proc. of CALC '01, LNCS. Springer-Verlag, 2001.

15. N. A. Howgrave-Graham, *Computational Mathematics Inspired by RSA*. PhD thesis, University of Bath, 1998.
16. E. Jochemz, A. May, *A Strategy for Finding Roots of Multivariate Polynomials with New Applications in Attacking RSA Variants*. In Advances in Cryptology (Asiacrypt 2006), Lecture Notes in Computer Science, Springer-Verlag, 2006.
17. A.K. Lenstra, H.W. Lenstra, Jr., and L. Lovász, *Factoring polynomials with rational coefficients*. Mathematische Ann., 261:513-534, 1982.
18. A. May, *Cryptanalysis of Unbalanced RSA with Small CRT-Exponent*. In Advances in Cryptology (Crypto 2002), Lecture Notes in Computer Science Volume 2442, pages 242-256, Springer Verlag, 2002.
19. A. May, *Computing the RSA Secret Key is Deterministic Polynomial Time Equivalent to Factoring*, In Advances in Cryptology (Crypto 2004), Lecture Notes in Computer Science Volume 3152, pages 213-219, Springer Verlag, 2004.
20. A. May, *Secret Exponent Attacks on RSA-type Schemes with Moduli $N = p^r q$*. In Practice and Theory in Public Key Cryptography (PKC 2004), Lecture Notes in Computer Science Volume 2947, pages 218-230, Springer-Verlag, 2004.
21. P.Q. Nguyen and D. Stehlé, *Floating-Point LLL Revisited*, Proceedings of Eurocrypt 2005. LNCS vol. 3494, Springer-Verlag, 2005.
22. V. Shoup, *Number Theory C++ Library (NTL) version 5.4*. Available at `www.shoup.net`.
23. V. Shoup, *OAEP reconsidered*. Proceedings of Crypto '01, vol. 2139, Lecture Notes in Computer Science.

A Proof of Lemma 1

Let R be a matrix basis of L and let U be the unimodular matrix such that :

$$U \cdot M = \begin{bmatrix} R \\ 0 \end{bmatrix}$$

(a unimodular matrix U satisfies $\det U = \pm 1$). Let V be the unimodular matrix such that $M' = M \cdot V$. Then :

$$U \cdot M \cdot V = U \cdot M' = \begin{bmatrix} R' \\ 0 \end{bmatrix},$$

where $R' = R \cdot V$ is a matrix basis for L'. Then

$$\det L' = |\det R'| = |\det(R \cdot V)| = |\det R| \cdot |\det V| = |\det R| = \det L.$$

B Proof of Lemma 3

The proof is very similar to the proof of Lemma 3 in [7]. It consists in showing that a matrix related to S is diagonally dominant, which enables to derive a lower bound for its determinant.

Let $W = \max_{i,j} |p_{ij}| X^i Y^j$ and let indices (u, v) such that $W = |p_{uv}| X^u Y^v$. Let indices (i_0, j_0) that maximize the quantity

$$8^{(i-u)^2+(j-v)^2} |p_{ij}| X^i Y^j.$$

The matrix S is obtained by taking the coefficients of the polynomials $x^a y^b p(x,y)$ for $0 \leq a,b < k$, taking only the coefficients of monomials $x^{i_0+i}y^{j_0+j}$ for $0 \leq i,j < k$. We must show :

$$\left(\frac{W}{X^{i_0}Y^{j_0}}\right)^{k^2} 2^{-6k^2\delta^2-2k^2} \leq |\det S| \leq \left(\frac{W}{X^{i_0}Y^{j_0}}\right)^{k^2} 2^{k^2} \tag{12}$$

We let $\mu(i,j) = ki+j$ be an index function; the matrix element $S_{\mu(a,b),\mu(i,j)}$ is the coefficient of $x^{i_0+i}y^{j_0+j}$ in $x^a y^b p(x,y)$, namely :

$$S_{\mu(a,b),\mu(i,j)} = p_{i_0+i-a,j_0+j-b}$$

We multiply each $\mu(i,j)$ column of S by

$$8^{2(i_0-u)i+2(j_0-v)j}X^{i_0+i}Y^{j_0+j}$$

and we multiply each $\mu(a,b)$ row by

$$8^{-2(i_0-u)a-2(j_0-v)b}X^{-a}Y^{-b}$$

to create a new matrix S' whose element is :

$$S'_{\mu(a,b),\mu(i,j)} = p_{i_0+i-a,j_0+j-b}X^{i_0+i-a}Y^{j_0+j-b}8^{2(i_0-u)(i-a)+2(j_0-v)(j-b)}$$

and we have :

$$\det S' = \det S \cdot \left(X^{i_0}Y^{j_0}\right)^{k^2} \tag{13}$$

Now we show that S' is a diagonally dominant matrix. Let denote $\tilde{p}_{ij} = p_{ij}X^iY^j$; the elements of matrix S' are :

$$S'_{\mu(a,b),\mu(i,j)} = \tilde{p}_{i_0+i-a,j_0+j-b}8^{2(i_0-u)(i-a)+2(j_0-v)(j-b)}$$

From maximality of (i_0,j_0) we have :

$$|\tilde{p}_{i_0+i-a,j_0+j-b}| \cdot 8^{(i-a+i_0-u)^2+(j-b+j_0-v)^2} \leq |\tilde{p}_{i_0j_0}|8^{(i_0-u)^2+(j_0-v)^2}$$

which gives :

$$|\tilde{p}_{i_0+i-a,j_0+j-b}| \cdot 8^{2(i-a)(i_0-u)+2(j-b)(j_0-v)} \leq |\tilde{p}_{i_0j_0}|8^{-(i-a)^2-(j-b)^2}$$

and then :

$$|S'_{\mu(a,b),\mu(i,j)}| \leq |\tilde{p}_{i_0,j_0}|8^{-(i-a)^2-(j-b)^2}$$

Each diagonal element $S'_{\mu(a,b),\mu(a,b)}$ of matrix S' is equal to $\tilde{p}_{i_0,j_0}$, and using :

$$\sum_{(i,j)\neq(a,b)} 8^{-(i-a)^2-(j-b)^2} \leq \sum_{(i,j)\neq(0,0)} 8^{-i^2-j^2} \leq -1 + \sum_{(i,j)} 8^{-i^2-j^2}$$
$$\leq -1 + \left(\sum_i 8^{-i^2}\right)^2 \leq \frac{3}{4}$$

we obtain that the sum of the absolute values of the off-diagonal entries in each $\mu(a,b)$ row is at most $\frac{3}{4}|\tilde{p}_{i_0,j_0}|$. Therefore matrix S' is diagonally dominant and each eigenvalue λ must verify :

$$\frac{1}{4}|\tilde{p}_{i_0,j_0}| \leq |\lambda| \leq \frac{7}{4}|\tilde{p}_{i_0,j_0}|$$

which gives :

$$|\tilde{p}_{i_0,j_0}|^{k^2} 2^{-2k^2} \leq |\det S'| \leq |\tilde{p}_{i_0,j_0}|^{k^2} 2^{k^2} \tag{14}$$

From the optimality of (i_0, j_0), we have :

$$8^{(i_0-u)^2+(j_0-v)^2}|\tilde{p}_{i_0,j_0}| \geq 8^{0+0}|\tilde{p}_{u,v}| = W$$

which gives :

$$8^{-2\delta^2} W \leq |\tilde{p}_{i_0,j_0}| \leq W$$

Combining with (14) we obtain :

$$W^{k^2} 2^{-6k^2\delta^2-2k^2} \leq |\det S'| \leq W^{k^2} \cdot 2^{k^2}$$

and using (13) we obtain (12).