J. Cryptology (1997) 10: 233–260

Journal of
CRYPTOLOGY

Small Solutions to Polynomial Equations, and Low Exponent RSA Vulnerabilities

Don Coppersmith
IBM Research, T. J. Watson Research Center,
Yorktown Heights, NY 10598, U.S.A.

Communicated by Andrew M. Odlyzko

Received 21 December 1995 and revised 11 August 1996

Abstract. We show how to find sufficiently small integer solutions to a polynomial in a single variable modulo N, and to a polynomial in two variables over the integers. The methods sometimes extend to more variables. As applications: RSA encryption with exponent 3 is vulnerable if the opponent knows two-thirds of the message, or if two messages agree over eight-ninths of their length; and we can find the factors of $N = PQ$ if we are given the high order $\frac{1}{4}\log_2 N$ bits of P.

Key words. Polynomial, RSA, Factoring.

1. Introduction

It is easy to compute the integer roots of a polynomial in a single variable over the integers

$$p(x) = 0.$$

But two related problems can be hard:

(1) finding integer roots of a *modular* polynomial in one variable:

$$p(x) = 0 \pmod{N};$$

(2) finding integer roots of a polynomial in *several* variables:

$$p(x, y) = 0.$$

In this paper we restrict these problems to the case where there exists a solution small enough (with respect to N or to the coefficients of p), and we can solve the problems in these special cases, using lattice basis reduction techniques.

Let N be a large composite integer of unknown factorization. Let

$$p(x) = x^{\delta} + p_{\delta-1}x^{\delta-1} + \cdots + p_2x^2 + p_1x + p_0$$

be a monic integer polynomial of degree δ in a single variable x. Suppose there is an integer solution x_0 to

$$p(x_0) = 0 \pmod{N}$$

satisfying

$$|x_0| < N^{1/\delta}.$$

We will show how to find such a solution x_0, in time polynomial in $(\log N, 2^\delta)$.

Suppose next that

$$p(x, y) = \sum_{ij} p_{ij} x^i y^j$$

is an irreducible integer polynomial in two variables over the integers (not modulo N this time), with degree δ in each variable separately. Let X and Y be upper bounds on the desired integer solution (x_0, y_0), and set

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

We will find an integer solution (x_0, y_0) satisfying $p(x_0, y_0) = 0$ if one exists with $|x_0| \leq X$, $|y_0| \leq Y$, provided

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

The techniques used in the two cases are similar. We use the coefficients of the polynomial p to build a matrix M, whose rows give the basis of an integer lattice. We will consider a row vector $\mathbf{r}$ whose entries are powers of the desired solutions: x_0^i or $x_0^i y_0^j$. The vector $\mathbf{s} = \mathbf{r}M$ will be a relatively short lattice element. Using lattice basis reduction techniques such as those due to Lovász [9] to analyze M, we find a hyperplane containing all the short lattice elements. The equation of this hyperplane translates to a linear relation on the elements of $\mathbf{r}$, and then to a polynomial equation $c(x_0) = 0$ or $c(x_0, y_0) = 0$ over $\mathbf{Z}$. In the univariate modular case we solve $c(x_0) = 0$ directly for x_0. In the bivariate integer case we combine $c(x_0, y_0)$ with $p(x_0, y_0)$ and solve.

An important application of the univariate modular case is to RSA encryption [12] with small exponent, when most of the message is fixed or “stereotyped.” Suppose the plaintext m consists of two pieces, a known piece B and an unknown piece x: $m = B + x$. Suppose m is RSA-encrypted with an exponent of 3, so the ciphertext c is given by $c = m^3 = (B + x)^3 \pmod{N}$. If we know B, c, and N, we can apply the present results to the modular polynomial equation

$$p(x) = (B + x)^3 - c = 0 \pmod{N},$$

and recover x as long as $|x| < N^{1/3}$, that is, x has fewer than one-third of the bits of the message, and these bits are consecutive.

A second application of the univariate modular case to RSA encryption with small exponent concerns random padding. Suppose a message m is padded with a random value r_1 before encrypting with exponent 3, giving the ciphertext

$$c_1 = (m + r_1)^3 \pmod{N}.$$

Suppose m is encrypted again with different random padding:

$$c_2 = (m + r_2)^3 \pmod N.$$

We will show how to recover m from c_1, c_2, N, as long as the random padding r_i is less than one-ninth of the bits of N. This is completely different from Hastad's [7] attack on low-exponent RSA; he used encryptions under several different moduli, and we use only one modulus.

The bivariate integer case can be applied to the problem of factoring an integer when we know its high-order bits. If we know $N = PQ$ and we know the high-order $\frac{1}{4}\log_2 N$ bits of P, then by solving the equation $(P_0 + x)(Q_0 + y) - N$ over a suitable range of x and y we can find the factorization of N. By comparison, Rivest and Shamir [13] need about $\frac{1}{3}\log_2 N$ bits of P, and a recent work of the present author [4] required $\frac{3}{10}\log_2 N$ bits. This has applications to some RSA-based cryptographic schemes; see, for example, Vanstone and Zuccherato [15].

The rest of the paper is organized as follows. In Section 2 we recall the necessary facts about lattice basis reduction. In Section 3 we present a heuristic approach, which does not quite work, but whose ideas will be refined in the present work. For the univariate modular case, we show in Section 4 how to build the matrix M, the rows of which generate our lattice. In Section 5 we analyze the determinant of this matrix, and compare to the length of the relevant vector. We complete the solution of the modular univariate polynomial in Section 6. Applications to RSA encryption with low exponent and partial information are given in Section 7 (where most of a message is known beforehand) and Section 8 (where two messages agree over most of their length). In Section 10 we develop the bivariate integer case, and apply it in Section 11 to the problem of factoring integers with partial information. Section 12 investigates the extension of these results to two or more variables modulo N or three or more variables over the integers. We give concluding remarks and an open problem in Section 13.

This paper, containing material from the author's papers [2] and [3], grew out of the joint work with Franklin, Patarin, and Reiter [5], which in turn was inspired Franklin and Reiter's Crypto '95 rump session talk [6].

2. Lattice Basis Reduction

We recall here some basic facts about lattice basis reduction. The reader is referred to [9] for further information.

Suppose M is a square $n \times n$ matrix with rational entries and with full rank. The rows of M generate a *lattice* L, a collection of vectors closed under addition and subtraction; in fact the rows form a *basis* of L.

From [9] we learn how to compute a *reduced* basis $(\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n)$ for L. The matrix B with rows $\mathbf{b}_i$ is related to M by a series of elementary row operations; equivalently, $B = KM$ where K is an invertible matrix, and both K and K^{-1} have integer entries. The computation of B is done in time polynomial in n and in $\log(\max\{|n_{ij}|, |d_{ij}|\})$, where n_{ij} and d_{ij} are the numerator and denominator of the matrix element $M_{i,j}$ in lowest terms.

Remark. Lattice reduction works more efficiently with integer entries, but our lattice is easier to describe with rational entries. Converting between the two is not difficult.

The basis elements $\mathbf{b}_i$ are relatively short. The Euclidean norm $|\mathbf{b}_1|$ is within a multiplicative factor of $2^{(n-1)/2}$ of the norm of the smallest nonzero lattice element, and similar estimates hold for the other $\mathbf{b}_i$. Setting $D = |\det(M)| = |\det(B)|$, we have

$$D \leq \prod |\mathbf{b}_i| \leq 2^{n(n-1)/4} D.$$

The first inequality is Hadamard's inequality. The second is a property of the reduced basis; see [9, equation (1.8)].

Let $\mathbf{b}_i^*$ denote the component of $\mathbf{b}_i$ orthogonal to the span of $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_{i-1}$. We know that

$$D = \prod |\mathbf{b}_i^*|.$$

From the discussion in [9] we know that the last basis element $\mathbf{b}_n$ satisfies

$$|\mathbf{b}_n^*| \geq D^{1/n} 2^{-(n-1)/4}.$$

(Note the direction of the inequality.) This follows from $|\mathbf{b}_i^*|^2 \leq 2|\mathbf{b}_{i+1}^*|^2$ and $D = \prod |\mathbf{b}_i^*|$.

Each lattice element $\mathbf{s}$ can be expressed as $\mathbf{s} = \sum s_i \mathbf{b}_i$, where the s_i are integers. Further, $|\mathbf{s}| \geq |s_n| \times |\mathbf{b}_n^*|$. So if $\mathbf{s}$ satisfies $|\mathbf{s}| < |\mathbf{b}_n^*|$, then s_n must be 0, and $\mathbf{s}$ must lie in the hyperplane spanned by $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_{n-1}$. Thus we have proved:

Lemma 1. *If a lattice element* $\mathbf{s}$ *satisfies* $|\mathbf{s}| < D^{1/n} 2^{-(n-1)/4}$ *then* $\mathbf{s}$ *lies in the hyperplane spanned by* $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_{n-1}$.

In our applications, we are not necessarily looking for the shortest nonzero vector in the lattice, but for a relatively short vector, and Lemma 1 serves to confine all such short vectors to a hyperplane. Lemma 2 generalizes this concept from a hyperplane to a subspace of smaller dimension.

Lemma 2. *If a lattice element* $\mathbf{s}$ *satisfies* $|\mathbf{s}| < |\mathbf{b}_i^*|$ *for all* $i = k+1, \ldots, n$, *then* $\mathbf{s}$ *lies in the space spanned by* $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_k$.

Lemma 2 will be useful when we wish to develop more than one equation. This will be necessary when solving a modular polynomial with more than one variable, or an integer polynomial with more than two variables. See Section 12.

3. Motivation

Lattice reduction techniques seem inherently linear. It is not immediately obvious how to apply these techniques to the nonlinear problem of solving polynomial equations.

To motivate the present work, we start with a heuristic approach to solving a modular polynomial equation by lattice basis reduction techniques. This approach does not quite work, but it gives ideas upon which we can build the algorithms which do work.

Given a monic univariate modular polynomial equation

$$p(x) = x^\delta + p_{\delta-1}x^{\delta-1} + \cdots + p_2x^2 + p_1x + p_0 = 0 \pmod N$$

to which we wish to find a small root x_0, we could proceed as follows:

Establish a suitable upper bound X on the size of the desired root x_0. Build a $(\delta + 2) \times (\delta + 2)$ matrix M, with diagonal elements given by $1, X^{-1}, X^{-2}, \ldots, X^{-\delta}, N$, and with right-hand column $p_0, p_1, p_2, \ldots, p_{\delta-1}, p_\delta = 1, N$; all other entries are 0.

$$M = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & p_0 \\ 0 & X^{-1} & 0 & \cdots & 0 & p_1 \\ 0 & 0 & X^{-2} & \cdots & 0 & p_2 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & X^{-\delta} & p_\delta \\ 0 & 0 & 0 & \cdots & 0 & N \end{bmatrix}.$$

Suppose $p(x_0) = y_0N$ for unknown integers x_0 and y_0 with $|x_0| < X$. Consider the row vector $\mathbf{r}$ consisting of powers of x_0 and y_0:

$$\mathbf{r} = (1, x_0, x_0^2, \ldots, x_0^{\delta-1}, x_0^\delta, -y_0).$$

Consider the row vector

$$\mathbf{s} = \mathbf{r}M = \left(1, \left(\frac{x_0}{X}\right), \left(\frac{x_0}{X}\right)^2, \ldots, \left(\frac{x_0}{X}\right)^{\delta-1}, \left(\frac{x_0}{X}\right)^\delta, 0\right).$$

Its last element is $p(x_0) - y_0N = 0$. The vector $\mathbf{s}$ is an element of the lattice spanned by the rows of M. Its Euclidean norm is bounded by $\sqrt{\delta + 1}$ since each entry $(x_0/X)^i$ is bounded by 1. If $\mathbf{s}$ is among the shorter vectors of this lattice, we might find it by lattice basis reduction techniques.

From the discussion in Section 2, we need to compare $|\mathbf{s}|$ to the $\delta + 2$ root of the determinant of the matrix M. If

$$|\mathbf{s}| < |\det(M)|^{1/(\delta+2)},$$

then $\mathbf{s}$ will be among the shorter vectors, and the lattice basis reduction techniques might find it. (For the present discussion we ignore factors like $2^{-(n-1)/4}$ dependent only on the size of the matrix. We will take account of them later.)

We can easily evaluate $\det(M)$ because M is upper triangular:

$$\det(M) = (1)(X^{-1})(X^{-2}) \cdots (X^{-\delta})N = NX^{-\delta(\delta+1)/2}.$$

Ignoring factors like $2^{-(n-1)/4}$ and $\delta + 1$, we require roughly that $\det(M) > 1$, and so we require roughly that

$$X^{(\delta^2+\delta)/2} < N,$$

$$X < N^{2/(\delta^2+\delta)},$$

quite a small bound on X, especially for moderately large values of δ. By contrast, the present paper will develop a more reasonable bound of (roughly)

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

One problem with this heuristic approach is that, although the entries r_i of the vector $\mathbf{r}$ are supposed to represent powers of x_0, there is no way (within the lattice structure) to enforce that relationship, for example, to enforce the requirement $r_{i+1}/r_i = r_{j+1}/r_j$.

A second, related, problem is that we have many unknowns r_i and only one relation $p(x_0) = y_0 N$. Each unknown r_i contributes a factor X^{-i} to $\det(M)$, and the lone relation $p(x_0) = y_0 N$ contributes a factor N. The resulting imbalance, and the requirement $\det(M) > 1$, lead to the stringent requirement $X^{\delta(\delta+1)/2} < N$.

In the new approach we will work with several relations: for example, $x_0^i p(x_0)^j = 0 \pmod{N^j}$. This allows us to reuse the unknowns r_i and amortize their "cost" over the several relations. Each relation, meanwhile, contributes a factor of N^j to $\det(M)$. Because $\det(M)$ now contains several powers of N, the requirement $\det(M) > 1$ translates to a much looser requirement on X.

The fact that the equations $x_0^i p(x_0)^j = 0 \pmod{N^j}$ hold mod N^j (rather than just mod N) improves this situation, by contributing larger powers of N to $\det(M)$. Using only equations of the form $x^i p(x) = 0 \pmod N$, we could find solutions x_0 up to about $X' = N^{1/(2\delta-1)}$. With the additional equations $x^i p(x)^j = 0 \pmod{N^j}$, we are able to improve this bound to $X = N^{1/\delta}$.

Notice that the r_i satisfy several equations that differ only by shifts in the powers of x_0. If $p(x) = x^3 + Ax^2 + Bx + C$, then two equations derived from $p(x_0) = 0 \pmod N$ and $x_0 p(x_0) = 0 \pmod N$ are

$$\begin{aligned} r_3 + Ar_2 + Br_1 + Cr_0 &= 0 \pmod N, \\ r_4 + Ar_3 + Br_2 + Cr_1 &= 0 \pmod N. \end{aligned}$$

The present approach allows us to recapture the flavor of the requirement that the various r_i should be related by (for example) $r_3/r_2 = r_4/r_3$, since the roles played by r_3 and r_2 in the first equation are the same as the roles played by r_4 and r_3 in the second equation. This is offered only as an intuitive explanation for the success of the present approach; it will not be used in the technical discussions that follow.

The use of Lemma 1 allows a qualitative innovation in the application of lattice basis reduction techniques, which may be of interest in its own right. We can state with certainty that the present algorithm will find all sufficiently small solutions, in all cases; by contrast, many applications of lattice basis reduction techniques can only be guaranteed to work in a large proportion of problem instances. By looking at the last element of the reduced basis (rather than the first), we can confine *all* sufficiently short lattice elements to a hyperplane whose equation we compute. In particular, the relatively short vector $\mathbf{s}$, corresponding to the desired solution, lies in this hyperplane. The equation of that hyperplane, together with the interpretation $r_i = x_0^i$, gives a polynomial equation which x_0 is guaranteed to satisfy. This guarantee is a new aspect of the present work.

4. Building the Matrix: Univariate Modular Case

In this section we show how to build the appropriate lattice for the case of a univariate modular polynomial. N is a large composite integer of unknown factorization. We are given the polynomial

$$p(x) = x^{\delta} + p_{\delta-1}x^{\delta-1} + \cdots + p_2x^2 + p_1x + p_0 = 0 \pmod N,$$

which we assume to be monic, that is, $p_{\delta} = 1$.

Suppose there is an integer x_0 satisfying

$$p(x_0) = 0 \pmod N$$

with

$$|x_0| < \frac{N^{(1/\delta)-\varepsilon}}{2}$$

for some $\varepsilon > 0$. We wish to find x_0.

Begin by selecting an integer

$$h \geq \max\left(\frac{\delta - 1 + \varepsilon\delta}{\varepsilon\delta^2}, \frac{7}{\delta}\right).$$

The first condition ensures that

$$\frac{h-1}{h\delta - 1} \geq \frac{1}{\delta} - \varepsilon.$$

The second condition ensures that $h\delta \geq 7$.

For each pair of integers i, j satisfying $0 \leq i < \delta$, $1 \leq j < h$, we define the polynomial

$$q_{ij}(x) = x^i p(x)^j.$$

For the desired solution x_0 we know that $p(x_0) = y_0N$ for some integer y_0, so that

$$q_{ij}(x_0) = 0 \pmod{N^j}.$$

We will build a rational matrix M of size $(2h\delta - \delta) \times (2h\delta - \delta)$, using the coefficients of the polynomials $q_{ij}(x)$, in such a way that an integer linear combination of the rows of M corresponding to powers of x and y will give a vector with relatively small Euclidean norm. Multiplying by least common denominator will produce an integer matrix on which lattice basis reduction can be applied.

The matrix M is broken into blocks. The upper right block, of size $(h\delta) \times (h\delta - \delta)$, has rows indexed by the integer g with $0 \leq g < h\delta$, and columns indexed by $\gamma(i, j) = h\delta + i + (j-1)\delta$ with $0 \leq i < \delta$ and $1 \leq j < h$, so that $h\delta \leq \gamma(i, j) < 2h\delta - \delta$. The entry at $(g, \gamma(i, j))$ is the coefficient of x^g in the polynomial $q_{ij}(x)$.

The lower right $(h\delta - \delta) \times (h\delta - \delta)$ block is a diagonal matrix, with the value N^j in each column $\gamma(i, j)$.

The upper left $(h\delta) \times (h\delta)$ block is a diagonal matrix, whose value in row g is a rational approximation to $X^{-g}/\sqrt{h\delta}$, where $X = \frac{1}{2}N^{(1/\delta)-\varepsilon}$ is an upper bound to the solutions $|x|$ of interest.

The lower left $(h\delta - \delta) \times (h\delta)$ block is zero.

We illustrate the matrix M in the case $h = 3$, $\delta = 2$. Assume that $p(x) = x^2 + ax + b$ and $p(x)^2 = x^4 + cx^3 + dx^2 + ex + f$. For simplicity we write τ instead of $1/\sqrt{h\delta}$.

$$M = \begin{bmatrix} \tau & 0 & 0 & 0 & 0 & 0 & b & 0 & f & 0 \\ 0 & \tau X^{-1} & 0 & 0 & 0 & 0 & a & b & e & f \\ 0 & 0 & \tau X^{-2} & 0 & 0 & 0 & 1 & a & d & e \\ 0 & 0 & 0 & \tau X^{-3} & 0 & 0 & 0 & 1 & c & d \\ 0 & 0 & 0 & 0 & \tau X^{-4} & 0 & 0 & 0 & 1 & c \\ 0 & 0 & 0 & 0 & 0 & \tau X^{-5} & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & N & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & N & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & N^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & N^2 \end{bmatrix}.$$

The rows of M span a lattice. Of interest to us is one vector $\mathbf{s}$ in that lattice, related to the unknown solution x_0. Consider a row vector $\mathbf{r}$ whose left-hand elements are powers of x_0:

$$r_g = x_0^g,$$

and whose right-hand elements are the negatives of powers of x_0 and y_0:

$$r_{\gamma(i,j)} = -x_0^i y_0^j,$$

$$\mathbf{r} = (1, x_0, x_0^2, \ldots, x_0^{h\delta-1}, -y_0, -x_0 y_0, \ldots, -x_0^{\delta-1} y_0, -y_0^2, -x_0 y_0^2, \ldots, -x_0^{\delta-1} y_0^{h-1}).$$

The product $\mathbf{s} = \mathbf{r}M$ is a row vector with left-hand elements given by

$$s_g = \frac{(x_0/X)^g}{\sqrt{h\delta}}$$

and right-hand elements by

$$s_{\gamma(i,j)} = q_{ij}(x_0) - x_0^i y_0^j N^j = 0.$$

The Euclidean norm of $\mathbf{s}$ is estimated by

$$|\mathbf{s}| = \left[\sum_g s_g{}^2\right]^{1/2} < \left[\sum_g \left(\frac{1}{\sqrt{h\delta}}\right)^2\right]^{1/2} = 1.$$

Because the right-hand elements $h\delta - \delta$ of the desired vector $\mathbf{s}$ are 0, we can restrict our attention to the sublattice $\hat{M}$ of M consisting of points with right-hand elements 0, namely $M \cap (\mathbf{R}^{h\delta} \times \{0\}^{h\delta-\delta})$. To do this computationally, we take advantage of the fact that $p(x)$ and hence $q_{ij}(x)$ are monic polynomials, so that certain $h\delta - \delta$ rows of the upper right block of M form an upper triangular matrix with 1 on the diagonal.

This implies that we can do elementary row operations on M to produce a block matrix $\tilde{M}$ whose lower-right $(h\delta - \delta) \times (h\delta - \delta)$ block is the identity and whose upper-right $(h\delta) \times (h\delta - \delta)$ block is zero.

The upper-left $(h\delta) \times (h\delta)$ block $\hat{M}$ of $\tilde{M}$ represents the desired sublattice: an $h\delta$-dimensional lattice, of which $\mathbf{s}$ is one relatively short element.

5. Analysis of the Determinant

M is an upper triangular matrix, so its determinant is just the product of the diagonal elements:

$$\begin{aligned}\det(M) &= \prod_g \frac{1}{X^g \sqrt{h\delta}} \prod_{ij} N^j \\ &= \frac{N^{\delta h(h-1)/2} X^{-(h\delta)(h\delta-1)/2}}{\sqrt{h\delta}^{h\delta}} \\ &= [N^{(h-1)/2} X^{-(h\delta-1)/2} (h\delta)^{-1/2}]^{h\delta}.\end{aligned}$$

By construction,

$$\det(M) = \det(\tilde{M}) = \det(\hat{M}) \times \det(I) = \det(\hat{M}).$$

We will be invoking Lemma 1 on the smaller matrix $\hat{M}$, whose dimension is $n = h\delta$. Since we know

$$|\mathbf{s}| < 1,$$

the required condition is

$$1 \leq |\det(\hat{M})|^{1/h\delta} 2^{-(h\delta-1)/4}.$$

Since

$$\det(\hat{M}) = (N^{(h-1)/2} X^{-(h\delta-1)/2} (h\delta)^{-1/2})^{h\delta},$$

this holds if

$$1 \leq N^{(h-1)/2} X^{-(h\delta-1)/2} (h\delta)^{-1/2} 2^{-(h\delta-1)/4},$$

that is, if

$$X \leq N^{(h-1)/(h\delta-1)} (h\delta)^{-1/(h\delta-1)} 2^{-1/2}.$$

So the hypothesis of Lemma 1 will hold if

$$X \leq N^{(h-1)/(h\delta-1)} (h\delta)^{-1/(h\delta-1)} 2^{-1/2}.$$

By our choice of h we have $h\delta \geq 7$, so that (by a computation)

$$(h\delta)^{-1/(h\delta-1)} > 2^{-1/2}.$$

Also by our choice of h we know

$$\frac{h-1}{h\delta - 1} \geq \frac{1}{\delta} - \varepsilon.$$

So if we select

$$X \leq \tfrac{1}{2} N^{(1/\delta)-\varepsilon},$$

we will have

$$|\mathbf{s}| < 1 \leq \det(\hat{M})^{1/n} 2^{-(n-1)/4},$$

as required.

6. Finishing the Solution

Now we can tie the pieces together.

Apply a lattice basis reduction routine to the row basis of the matrix $\hat{M}$, producing a basis $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n$, satisfying

$$|\mathbf{b}_n^*| \geq \det(\hat{M})^{1/n} 2^{-(n-1)/4},$$

where, as before, $n = h\delta = \dim(\hat{M})$.

By the calculation in the previous section, we have

$$|\mathbf{b}_n^*| \geq 1.$$

By Lemma 1, any vector in the lattice generated by the rows of $\hat{M}$ with length less than 1 must lie in the hyperplane spanned by $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_{n-1}$.

In terms of the larger matrix M and the vectors $\mathbf{r}, \mathbf{s}$ with $\mathbf{r}M = \mathbf{s}$, there is an $h\delta$-dimensional space of vectors $\mathbf{r}$ such that $\mathbf{r}M = \mathbf{s}$ has 0's in its right-hand $h\delta - \delta$ entries. By Lemma 1, those integer vectors $\mathbf{r}$ which additionally satisfy $|\mathbf{s}| < 1$ must lie in a space of dimension one smaller, namely dimension $h\delta - 1$. This gives rise to a linear equation on the entries r_g, $0 \leq g < h\delta$. That is, we compute coefficients c_g, not all zero, such that:

For any integer vector $\mathbf{r} = (r_g, r_{\gamma(i,j)})$ *such that* $\mathbf{s} = \mathbf{r}M$ *has right-hand entries* 0 *and* $|\mathbf{s}| < 1$, *we must have*

$$\sum c_g r_g = 0.$$

This holds for all short vectors $\mathbf{s}$ in the lattice with right-hand side 0. In particular, it holds for the vector obtained from $\mathbf{r}$ where

$$r_g = x_0^g, \qquad r_{\gamma(i,j)} = -x_0^i y_0^j.$$

Thus we have computed coefficients c_g of a polynomial $C(x)$ such that the small solution x_0 satisfies

$$C(x_0) = \sum c_g x_0^g = 0.$$

This is a polynomial equation holding in $\mathbf{Z}$, not just modulo N. We can solve this polynomial for x_0 easily, using known techniques for solving univariate polynomial equations over $\mathbf{Z}$. (The Sturm sequence [14] will suffice.) Thus we have produced the desired solution x_0.

Remark. If there are several short solutions x_0, this procedure will find all of them simultaneously. All will be roots of the polynomial

$$C(x_0) = \sum c_g x_0^g = 0.$$

We have proved:

Theorem 1. *Let $p(x)$ be a polynomial of degree δ in one variable modulo an integer N of unknown factorization. Let X be the bound on the desired solution x_0. If*

$$X < \tfrac{1}{2} N^{1/\delta - \varepsilon},$$

then in time polynomial in $(\log N, \delta, 1/\varepsilon)$, *we can find all integers x_0 with $p(x_0) = 0 \pmod{N}$ and $|x_0| < X$.*

Proof. The lattice basis reduction step operated on a matrix of size $h\delta = O(\delta/\varepsilon)$, and the matrix entries are not too large. By [9] this step is done in polynomial time. The rest of the algorithm is also polynomial time. □

Corollary 1. *With the hypothesis of Theorem* 1, *except that*

$$X \le N^{1/\delta},$$

then in time polynomial in $(\log N, 2^\delta)$, *we can find all integers x_0 such that $p(x_0) = 0 \pmod{N}$ and $|x_0| \le X$.*

Proof. Cover the interval $[-N^{1/\delta}, N^{1/\delta}]$ by four intervals I_i of length $\frac{1}{2}N^{1/\delta}$, each centered at some integer x_i. For each value i, apply Theorem 1 with $\varepsilon = 1/\log N$ to the polynomial $p_i(x) = p(x + x_i)$ to find all solutions $x_0 = x + x_i$ within the interval I_i, in time polynomial in $(\log N, 2^\delta)$. □

7. Application: Stereotyped Messages

An important application of the univariate modular case is to RSA encryption [12] with small exponent, when most of the message is fixed or "stereotyped." Suppose the plaintext m consists of two pieces:

(1) A known piece $B = 2^k b$, such as the ASCII representation of "October 19, 1995. The secret key for the day is."
(2) An unknown piece x, such as "Squeamish Ossifrage," whose length is less than one-third the length of N.

Suppose this is RSA-encrypted with an exponent of 3, so the ciphertext c is given by $c = m^3 = (B + x)^3 \pmod{N}$. If we know B, c and N, we can apply the present results to the polynomial $p(x) = (B + x)^3 - c$, and recover x_0 satisfying

$$p(x_0) = (B + x_0)^3 - c = 0 \pmod{N}$$

as long as such an x_0 exists with $|x_0| < N^{1/3}$, that is, the length of x_0 is less than one-third of the length of N.

This is obvious when $B = 0$: if the plaintext is just $x_0 < N^{1/3}$, then $x_0^3 < N$, and the ciphertext is $c = x_0^3$ as integers, so that we could recover $x_0 = c^{1/3}$ by taking the integer cube root. But the present paper makes it possible for nonzero B as well.

Remark. The bound X on recoverable values x_0 depends on the modulus N. If x_0 has 250 bits and N has 512 bits, and an RSA exponent of 3 is used, the present techniques fail to recover x_0 because $x_0 > N^{1/3}$. But if we upgrade to a 1024-bit modulus N while keeping the unknown x_0 at 250 bits, these x_0 are now vulnerable to attack because $x_0 < N^{1/3}$.

The attack works equally well if the unknown x_0 lies in the most significant bits of the message m rather than the least significant bits—we are just multiplying x by a known constant 2^k.

An interesting variant occurs when the unknown x is split between two blocks:

“TODAY’S KEY IS swordfish AND THE PASSWORD IS joe.”

We can view this as two unknowns: x = “swordfish” and y = “joe,” and one known piece B = “TODAY’S KEY IS ——— AND THE PASSWORD IS —,” presuming that we know (or correctly guess) the lengths of x and y. The plaintext message is

$$m = B + 2^k x + y,$$

the ciphertext is

$$c = m^3 \pmod N,$$

and the polynomial which we wish to solve is

$$p(x, y) = c - (B + 2^k x + y)^3 = 0 \pmod N,$$

with a solution (x_0, y_0) suitably bounded.

We defer consideration of this case until Section 12.

8. Application to RSA with Random Padding: Two Messages

To introduce the second application (which was actually the starting point of the present investigation), we recall the recent result of Franklin and Reiter [6].

Suppose two messages m and m' satisfy a *known* affine relation, say

$$m' = m + r$$

with r known. Suppose we know the RSA-encryptions of the two messages with an exponent of 3:

$$\begin{aligned} c &= m^3 \pmod N, \\ c' &= (m')^3 = m^3 + 3m^2 r + 3mr^2 + r^3 \pmod N. \end{aligned}$$

Then we can recover m from c, c', r, and N:

$$m = \frac{r(c' + 2c - r^3)}{c' - c + 2r^3} = \frac{r(3m^3 + 3m^2r + 3mr^2)}{3m^2r + 3mr^2 + 3r^3} \pmod{N}.$$

What if we do not know the exact relation between m and m', but we do know that r is small, say

$$\begin{aligned} m' &= m + r, \\ |r| &< N^{1/9}. \end{aligned}$$

Can we still find m?

One can imagine a protocol in which messages M are subjected to random padding before being RSA-encrypted with an exponent of 3. Perhaps M is left-shifted by k bits, and a random k-bit quantity R is added, to form a plaintext m; the ciphertext c is then the cube of m (mod N):

$$c = m^3 = (2^k M + R)^3 \pmod{N}.$$

Now suppose the same unknown message M is encrypted twice, but with a different random pad each time. Let the second random pad be $R' = R + r$ so that the second plaintext is $m' = m + r$. Then we see the two ciphertexts

$$\begin{aligned} c &= m^3 = (2^k M + R)^3 \pmod{N}, \\ c' &= (m')^3 = (2^k M + R')^3 = (m + r)^3 \pmod{N}. \end{aligned}$$

Can we recover r and m, given knowledge of c, c', and N?

We can eliminate m from the two equations above by taking their resultant:

$$\begin{aligned} &\mathit{Resultant}_m(m^3 - c, (m + r)^3 - c') \\ &\quad = r^9 + (3c - 3c')r^6 + (3c^2 + 21cc' + 3(c')^2)r^3 + (c - c')^3 = 0 \pmod{N}. \end{aligned}$$

This is a univariate polynomial in r of degree 9 (mod N). If its solution r satisfies $|r| < N^{1/9}$, we can apply the present work to recover r. We can then apply Franklin and Reiter's result to recover m, and strip off the padding to get M.

As before, this works just as well if the padding goes in the high-order bits, or in the middle; just divide each plaintext by the appropriate power of 2 to move the random bits to the low-order bits.

The warning is clear: If the message is subject to random padding of length less than one-ninth the length of N, and then encrypted with an exponent of 3, multiple encryptions of the same message will reveal the message.

Notice that for a 1024-bit RSA key, this attack tolerates 100 bits of padding fairly easily.

Some possible steps to avoid this attack.

(1) Randomize the message in other ways; for example, by the methods of Bellare and Rogaway [1]. This spreads the randomization throughout the message in a nonlinear manner, and completely blocks the present attack.

(2) Spread the random padding into several blocks (not one contiguous block). Then the present attack needs to be modified. The padding could be two small blocks r and s, positioned so that the encryption is $c = (2^l r + 2^k m + s)^3 \pmod N$. Two encryptions of the same message would yield a resultant which is a single equation in two small integer variables r and s. The generalized attack of Section 12 might work, provided that $|r|$ and $|s|$ are subject to bounds R and S with $RS < N^{1/9}$. The computation is more complicated and results are not guaranteed.

(3) Spread the padding throughout the message: two bits out of each eight-bit byte, for example. This seems to be a much more effective defense against the present attack.

(4) Increase the amount of padding. This decreases efficiency; also if the padding is less than one-sixth the length of N, an alternate solution shown in Appendix 1 might still recover the message if multiple encryptions have been done.

(5) Make the "random" padding depend on the message deterministically. For example, we could subject the message to a hashing function, and append that hash value as the random padding. Then two encryptions would be identical, because the random padding would be identical. A possible weakness still exists: suppose a time-stamp is included in each message, and this time-stamp occupies the low-order bits, next to the padding. Then two plaintexts for the same message (with different time stamps) will differ in the time-stamp and the pad; just let r combine these two fields and proceed as before.

(6) Use larger exponents for RSA encryption. If the exponent is e, the attack apparently tolerates random padding of length up to $1/e^2$ times the length of N. So already for $e = 7$ the attack is useless: on a 1024-bit RSA key with $e = 7$, the attack would tolerate only 21 bits of padding, and this would be better treated by exhaustion.

9. RSA Signatures

The present work does not not show any weaknesses in the RSA signature scheme with a small validating exponent. For example, using the exponent $e = 3$, and using several related messages $m_i = m_0 + i$, $i = 0, 1, 2, \dots, 100$, the knowledge of signatures $m_i^{1/3} \pmod N$ for $m_0, m_1, \dots, m_{99}$ does not help us deduce the signature for m_{100}.

A crude analogy might illustrate the situation. Knowledge of the real cube roots $10^{1/3}, 11^{1/3}, 12^{1/3}, 13^{1/3}$ does not help us to compute $14^{1/3}$, since the five quantities are linearly independent over the rationals; in fact, $14^{1/3}$ is not in $\mathbf{Q}(10^{1/3}, 11^{1/3}, 12^{1/3}, 13^{1/3})$. But given the real cubes $10^3, 11^3, 12^3, 13^3$, we can easily compute 14^3 from

$$10^3 - 4 \times 11^3 + 6 \times 12^3 - 4 \times 13^3 + 14^3 = 0.$$

10. Bivariate Integer Case

We consider next the case of a single polynomial equation in two variables over the integers (not mod N):

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

for which we wish to find small integer solutions (x_0, y_0). We assume that $p(x, y)$ has maximum degree δ in each variable separately, and that $p(x, y)$ is irreducible over the integers. In particular, its coefficients are relatively prime as a set.

The basic outline is the same as before.

We create several polynomials

$$q_{ij}(x, y) = x^i y^j p(x, y)$$

satisfied by the desired solution (x_0, y_0), and build from these a matrix M representing a lattice. There will be a sublattice, represented by a smaller matrix $\hat{M}$, corresponding to vectors with right-hand side equal to 0. One vector $\mathbf{r}$ with entries $r_{gh} = x_0^g y_0^h$ will give rise to a short vector $\mathbf{s} = \mathbf{r}M$ in the sublattice. By lattice basis reduction techniques we confine all such short vectors to a hyperplane, whose equation,

$$\sum c_{gh} r_{gh} = 0,$$

for our special vector $\mathbf{r}$, translates to a polynomial equation on x_0 and y_0:

$$C(x_0, y_0) = \sum c_{gh} x_0^g y_0^h = 0.$$

We will see that $C(x, y)$ is not a multiple of $p(x, y)$, so that since p is irreducible, the resultant of C and p gives us enough information to find (x_0, y_0).

There are some technical differences between this bivariate integer case and the earlier univariate modular case.

In the modular case, we expressed the bound X in terms of the modulus N and the degree of p. Here, instead of N, we express bounds X and Y in terms of the coefficients of p. Define a polynomial $\tilde{p}(x, y) = p(xX, yY)$, so that $\tilde{p}_{ij} = p_{ij} X^i Y^j$. Define $W = \max_{ij} |\tilde{p}_{ij}|$ as the largest possible term in $p(x, y)$ in the region of interest. Then we will find a solution (x_0, y_0) bounded in absolute values by (X, Y) (if one exists) provided that

$$XY < W^{\{2/(3\delta)\}-\varepsilon}.$$

The matrices M_1 and $\hat{M}$ are rectangular rather than square, so that we are dealing with a k-dimensional lattice in $\mathbf{Z}^n$ with $k < n$. The lattice basis reduction routines handle this easily enough, but the quantity analogous to $\det(M)$ is harder to analyze in this case.

A minor difference is that we use polynomials $q_{ij}(x, y) = x^i y^j p(x, y)$ rather than $q_{ijk}(x, y) = x^i y^j p(x, y)^k$ to build our matrix M. It turns out that using powers of p would not help us, because we no longer gain the advantage that came from introducing moduli N^j instead of N.

We begin by selecting an integer $k > 2/(3\varepsilon)$.

For all pairs of integers (i, j) with $0 \leq i < k$ and $0 \leq j < k$, form the polynomial $q_{ij}(x, y) = x^i y^j p(x, y)$. Obviously $q_{ij}(x_0, y_0) = 0$.

Form a matrix M_1 with $(k + \delta)^2$ rows, indexed by $\gamma(g, h) = (k + \delta)g + h$ with $0 \leq g, h < k + \delta$. M_1 has $(k + \delta)^2 + k^2$ columns, the left-hand columns indexed by $\gamma(g, h)$ and the right-hand columns indexed by $\beta(i, j) = (k + \delta)^2 + ki + j$ with $0 \leq i, j < k$. The left-hand block is a diagonal matrix whose $(\gamma(g, h), \gamma(g, h))$ entry is given by $X^{-g} Y^{-h}$. The $(\gamma(g, h), \beta(i, j))$ entry of the right-hand block is the coefficient of $x^g y^h$ in the polynomial $q_{ij}(x, y)$.

Perform elementary row operations on M_1 to produce a matrix M_2 whose right-hand block has the $k^2 \times k^2$ identity matrix on the bottom and the $(2k\delta + \delta^2) \times k^2$ zero matrix on the top. We can do this because the greatest common divisor of the coefficients of p is 1 (p being irreducible). The lattice formed by these top $(2k\delta + \delta^2)$ rows of M_2 is the sublattice of the original lattice obtained by setting to 0 all the right-hand columns.

Now do lattice basis reduction on the top $2k\delta+\delta^2$ rows of M_2; let the resulting $2k\delta+\delta^2$ rows form a new matrix M_3.

Consider the $(k + \delta)^2$-long row vector $\mathbf{r}$ whose $\gamma(g, h)$ entry is $x_0^g y_0^h$. The row vector $\mathbf{s}$ of length $(k + \delta)^2 + k^2$ given by $\mathbf{s} = \mathbf{r}M_1$ satisfies

$$\begin{aligned} \mathbf{s}_{\gamma(g,h)} &= \left(\frac{x_0}{X}\right)^g \left(\frac{y_0}{Y}\right)^h, \\ |\mathbf{s}_{\gamma(g,h)}| &\le 1, \\ \mathbf{s}_{\beta(i,j)} &= q_{ij}(x_0, y_0) = 0, \\ |\mathbf{s}| &< k + \delta. \end{aligned}$$

Because its right-hand side is 0, $\mathbf{s}$ is one of the vectors in the row lattice spanned by M_3. We will show that it is a “relatively short” vector in the lattice. To do this, we need to estimate the sizes of the other vectors in M_3.

To that end, let M_4 be the matrix obtained from M_1 by multiplying the $\gamma(g, h)$ row by $X^g Y^h$ and multiplying the $\beta(i, j)$ column by $X^{-i}Y^{-j}$. So $M_4 = \Delta_1 M_1 \Delta_2$ where Δ_1 and Δ_2 are diagonal matrices. The left-hand block of M_4 is the $(k + \delta)^2 \times (k + \delta)^2$ identity matrix. Each column in the right-hand block represents the coefficients of the polynomial $x^i y^j p(xX, yY) = x^i y^j \tilde{p}(x, y)$: If $g = i + a$ and $h = j + b$, then

$$(M_1)_{\gamma(g,h),\beta(i,j)} = p_{ab},$$

$$(M_4)_{\gamma(g,h),\beta(i,j)} = p_{ab} X^g Y^h X^{-i} Y^{-j} = p_{ab} X^a Y^b = \tilde{p}_{ab}.$$

The right-hand columns are all shifted versions of one fixed column vector $\mathbf{v}$, representing the coefficients of the polynomial $\tilde{p}(x, y)$, namely

$$v_{\gamma(a,b)} = p_{ab} X^a Y^b = \tilde{p}_{ab}.$$

The largest element of each has absolute value W. These columns are selected columns of a Toeplitz matrix.

A lemma, whose proof is given in Appendix 2, says that these columns are nearly orthogonal.

Lemma 3. *There is a $k^2 \times k^2$ submatrix of M_4 with determinant at least*

$$W^{k^2} 2^{-6k^2\delta^2 - 2k^2}$$

(*in absolute value*). *If the largest coefficient of $\tilde{p}$ is one of $\tilde{p}_{00}$, $\tilde{p}_{0\delta}$, $\tilde{p}_{\delta 0}$, or $\tilde{p}_{\delta\delta}$, then the bound is W^{k^2}.*

The lemma finds a $k^2 \times k^2$ matrix of the right-hand block of M_4 with large determinant. Select $2k\delta + \delta^2$ columns of the left-hand block of M_4 (the identity matrix) to extend this to an $(k+\delta)^2 \times (k+\delta)^2$ submatrix of M_4 with the same determinant. Let T be the $((k+\delta)^2 + k^2) \times (k+\delta)^2$ permutation matrix selecting the appropriate $(k+\delta)^2 = (2k\delta + \delta^2) + (k^2)$ columns. So we have

$$|\det(\Delta_1 M_1 \Delta_2 T)| \geq W^{k^2} 2^{-6k^2\delta^2 - 2k^2}.$$

Now $\Delta_2 T = T\Delta_3$ where Δ_3 is a diagonal matrix differing from Δ_2 by the deletion of K 1's on the diagonal, so that

$$|\det(\Delta_1 M_1 T \Delta_3)| \geq W^{k^2} 2^{-6k^2\delta^2 - 2k^2}.$$

We compute the determinants of Δ_i:

$$\begin{aligned}\det(\Delta_1) &= \prod X^g Y^h = (XY)^{(k+\delta)^2(k+\delta-1)/2},\\ \det(\Delta_2) &= \det(\Delta_3) = \prod X^{-i} Y^{-j} = (XY)^{-k^2(k-1)/2}.\end{aligned}$$

Remark. Much cancellation goes on between $\det(\Delta_1)$ and $\det(\Delta_2)$: all the factors $X^i Y^j$ with $0 \leq i < k$ and $0 \leq j < k$ are cancelled, leaving only those factors $X^g Y^h$ with $(g,h) \in \{0,\ldots,k+\delta-1\}^2 - \{0,\ldots,k-1\}^2$. Thus the shape of the *boundary* of the region of applicable (g,h) is important, and must be considered when designing the algorithm.

Multiplying the two determinants, we get

$$\begin{aligned}\det(\Delta_1)\det(\Delta_2) &= (XY)^{[(k+\delta)^2(k+\delta-1)-k^2(k-1)]/2}\\ &= (XY)^{[3k^2\delta + k(3\delta^2-2\delta)+(\delta^3-\delta^2)]/2},\end{aligned}$$

and since

$$|\det(M_1 T)| \geq \frac{W^{k^2} 2^{-6k^2\delta^2-2k^2}}{\det(\Delta_1)\det(\Delta_2)},$$

we obtain

$$|\det(M_1 T)| \geq W^{k^2} 2^{-6k^2\delta^2-2k^2} (XY)^{-[3k^2\delta + k(3\delta^2-2\delta)+(\delta^3-\delta^2)]/2}.$$

Let this lower bound be called E. $M_3 T$ is obtained from $M_1 T$ by elementary row operations, so

$$|\det(M_3 T)| = |\det(M_1 T)| \geq E.$$

The row $\mathbf{s}T$ in $M_3 T$ obtained from $\mathbf{s}$ by deleting columns has Euclidean length bounded by that of $\mathbf{s}$:

$$|\mathbf{s}T| \leq |\mathbf{s}| < k + \delta.$$

$M_3 T$ has a block lower triangular structure, with a $k^2 \times k^2$ identity matrix in the lower right. Let $\hat{M}$ denote the upper-left block of $M_3 T$, with dimension $2k\delta + \delta^2$ on each side. We have

$$|\det(\hat{M})| = |\det(M_3 T)| \geq E.$$

We wish to apply Lemma 1 to $\hat{M}$ and $\mathbf{s}T$, with $n = 2k\delta + \delta^2$. If we can guarantee

$$k + \delta \le E^{1/n} 2^{-(n-1)/4},$$

then from

$$\begin{aligned} |\mathbf{s}T| &< k + \delta, \\ E &\le \det(\hat{M}), \end{aligned}$$

we will have

$$|\mathbf{s}T| < \det(\hat{M})^{1/n} 2^{-(n-1)/4}$$

as required by Lemma 1.

This requirement translates to

$$(k + \delta)^n \le E \times 2^{-n(n-1)/4}.$$

Recalling

$$\begin{aligned} n &= 2k\delta + \delta^2, \\ E &= W^{k^2} 2^{-6k^2\delta^2 - 2k^2} (XY)^{-[3k^2\delta + k(3\delta^2 - 2\delta) + (\delta^3 - \delta^2)]/2}, \end{aligned}$$

and omitting some tedious computations, we translate the requirement to

$$XY \le W^{2/3\delta - \varepsilon'} 2^{-(14\delta/3) - o(\delta)},$$

where

$$\varepsilon' \approx \frac{2}{3k}\left(1 - \frac{2}{3\delta}\right).$$

The rest of the construction proceeds as before. Assume XY satisfies this bound. Then from Lemma 1, applying lattice basis reduction to $\hat{M}$ will produce a hyperplane containing all lattice vectors as short as $\mathbf{s}T$. The equation of this hyperplane, and the construction of $\mathbf{s}$, yield the polynomial equation

$$C(x_0, y_0) = \sum c_{gh} x_0^g y_0^h = 0.$$

Further, $C(x, y)$ is not a multiple of $p(x, y)$, since all the multiples of $p(x, y)$ of sufficiently low degree were already used to define the sublattice $\hat{M}$. Since $p(x, y)$ is irreducible,

$$Q(x) = \mathit{Resultant}_y(C(x, y),\, p(x, y))$$

gives a nontrivial integer polynomial. We can easily compute its roots, which include x_0. Finally, given x_0, we can easily find those y solving $p(x_0, y) = 0$.

Tying this all together, we have:

Theorem 2. *Let $p(x, y)$ be an irreducible polynomial in two variables over $\mathbf{Z}$, of maximum degree δ in each variable separately. Let X, Y be bounds on the desired* *solutions* x_0, y_0. *Define* $\tilde{p}(x, y) = p(xX, yY)$ *and let* W *be the absolute value of the largest coefficient of* $\tilde{p}$. *If*

$$XY < W^{\{2/(3\delta)\}-\varepsilon}2^{-14\delta/3}$$

then in time polynomial in $(\log W, \delta, 1/\varepsilon)$, *we can find all integer pairs* (x_0, y_0) *with* $p(x_0, y_0) = 0$, $|x_0| < X$, *and* $|y_0| < Y$.

Proof. The lattice basis reduction step operated on a matrix of size $2k\delta + \delta^2$, where $k = O(1/\varepsilon)$. By [9] this step is done in polynomial time. The rest of the algorithm is also polynomial time. □

Corollary 2. *With the hypothesis of Theorem* 2, *except that*

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

then in time polynomial in $(\log W, 2^\delta)$, *we can find all integer pairs* (x_0, y_0) *with* $p(x_0, y_0) = 0$, $|x_0| \le X$, *and* $|y_0| \le Y$.

Proof. Set $\varepsilon = 1/\log W$, and do exhaustive search on the high-order $O(\delta)$ unknown bits of x. The running time is still polynomial, but of higher degree in $(\log W)$. □

Remark. Theorem 2 was developed for the case where p had degree δ independently in each variable. If the set of indices of nonzero coefficients of p (that is, its Newton polygon) has a different shape, it is useful to select the indices of the polynomials $q_{ij}(x, y)$ and monomials $x^g y^h$ in a different manner. The shape of the region of allowable monomials (g, h), and in particular its boundary, interacts with the shape of the Newton polygon of p in determining the efficiency of the algorithm.

Theorem 3. *With the hypothesis of Theorem* 2, *except that* p *has total degree* δ, *the appropriate bound is*

$$XY < W^{1/\delta}2^{-13\delta/2}.$$

Proof (Sketch). We use polynomials $q_{ij} = x^i y^j p(x, y)$ where $i + j < k$ (rather than $i < k$ and $j < k$ independently as before). The set of indices (i, j) now forms a triangle rather than a square. The relevant determinant is now

$$W^{k(k+1)/2}(XY)^{-\{(k+\delta+1)(k+\delta)(k+\delta-1)-(k+1)k(k-1)\}/6}2^{-3(2\delta^2)((1/2)k^2)-(k\delta)^2/4-o(k^2)}$$
$$= W^{k(k+1)/2}(XY)^{-\{3k^2\delta+3k\delta^2+\delta^3-\delta\}/6}2^{-3\delta^2k^2-(k\delta)^2/4-o(k^2)}.$$

Solve for XY to get

$$XY < W^{(1/\delta)-\varepsilon}2^{-13\delta/2},$$

where $\varepsilon = O(1/k)$. As in Corollary 2, set $\varepsilon = 1/\log W$ and exhaust on high-order bits while maintaining polynomial time. □

Remark. This shows how the shape of the region of indices (g, h) of monomials, and particularly the boundary of that shape, affects the outcome. Theorem 3 is better than Theorem 2 if p is a general polynomial of total degree δ, but Theorem 2 is better if p has degree δ in each variable independently.

As another example, if $p(x, y)$ has degree G in x and H in y (independently), then for any positive parameter α we can tolerate ranges X and Y satisfying

$$W >> X^{G+(\alpha H/2)} Y^{H+(G/(2\alpha))}$$

by allowing $0 \le i \le k\alpha$ and $0 \le j \le k$.

11. Factoring with High Bits Known

We can apply the present techniques to the problem of factoring an integer when we know the high-order bits of one of the factors.

Suppose we know $N = PQ$ and we know the high-order $\frac{1}{4}\log_2 N$ bits of P. By division we know the high-order $\frac{1}{4}\log_2 N$ bits of Q as well.

We write

$$\begin{aligned} P &= P_0 + x_0, \\ Q &= Q_0 + y_0, \end{aligned}$$

where P_0 and Q_0 are known, while P, Q, x_0, and y_0 are unknown. Define the bounds X and Y on the unknowns x_0 and y_0 by

$$\begin{aligned} |x_0| &< P_0 N^{-1/4} = X, \\ |y_0| &< Q_0 N^{-1/4} = Y. \end{aligned}$$

Define the polynomial

$$\begin{aligned} p(x, y) &= (P_0 + x)(Q_0 + y) - N \\ &= (P_0 Q_0 - N) + Q_0 x + P_0 y + xy, \end{aligned}$$

where x and y are dummy variables. One integer solution of $p(x, y) = 0$ is given by the desired (x_0, y_0), namely,

$$p(x_0, y_0) = PQ - N = 0.$$

We have $\delta = 1$, and the quantity W is given by

$$\begin{aligned} W &= \max_{ij}(|p_{ij}| X^i Y^j) \\ &= \max(|P_0 Q_0 - N|, Q_0 X, P_0 Y, XY) \\ &= N^{3/4}. \end{aligned}$$

An easy computation gives

$$\begin{aligned} XY &= P_0 Q_0 N^{-1/2} \approx N^{1/2} \\ &= W^{2/(3\delta)}, \end{aligned}$$

so that the hypothesis of Corollary 2 is satisfied. Thus we have:

Theorem 4. *In polynomial time we can find the factorization of $N = PQ$ if we know the high-order $(\frac{1}{4}\log_2 N)$ bits of P.*

By comparison, Rivest and Shamir [13] need about $(\frac{1}{3}\log_2 N)$ bits of P, and a recent paper by the present author [4] used a lattice-based method (less efficient than that of this paper) to factor N using $(\frac{3}{10}\log_2 N)$ bits of P.

Theorem 4 has applications to some RSA-based cryptographic schemes. For example, Vanstone and Zuccherato [15] design an ID-based RSA encryption scheme, where a person's identity is encoded in his RSA modulus $N = PQ$. In one variant of the scheme (Section 3.1 of [15]), a 1024-bit N is created by specifying (in a public manner) the high-order $512 - 248 = 264$ bits of P and hence of Q. By the present techniques, this is enough information to allow the attacker to factor N.

If we know the low-order bits of P instead of the high-order bits, we get the same results, but a twist in the proof is worth noticing.

Theorem 5. *In polynomial time we can find the factorization of $N = PQ$ if we know the low-order $(\frac{1}{4}\log_2 N)$ bits of P.*

Let $k = \lfloor \frac{1}{4}\log_2 N \rfloor$, so that

$$2^k \approx N^{1/4}.$$

Write

$$\begin{aligned} P &= 2^k x_0 + P_0, \\ Q &= 2^k y_0 + Q_0, \end{aligned}$$

where P_0 and Q_0 are known, while P, Q, x_0, and y_0 are unknown. Iterate over possible values of

$$\ell = \lceil \log_2(P) \rceil,$$

and define bounds X and Y by

$$\begin{aligned} |x_0| < X &= 2^{\ell-k} \approx PN^{-1/4}, \\ |y_0| < Y &= N2^{1-\ell-k} \approx QN^{-1/4}. \end{aligned}$$

Define the polynomial

$$\begin{aligned} p(x, y) &= [(2^k x + P_0)(2^k y + Q_0) - N]/2^k \\ &= 2^k xy + Q_0 x + P_0 y + [(P_0 Q_0 - N)/2^k], \end{aligned}$$

so that (x_0, y_0) is a root of the equation

$$p(x_0, y_0) = \frac{PQ - N}{2^k} = 0.$$

The term $(P_0 Q_0 - N)/2^k$ is an integer by construction. We needed to define $p(x, y)$ as above, rather than the apparent choice

$$p'(x, y) = (2^k x + P_0)(2^k y + Q_0) - N,$$

because the coefficients of p' all have the common factor 2^k, so that p' would be reducible over $\mathbf{Z}$, namely, $p' = 2^k \times p$, violating the hypothesis of Theorem 2. In particular, the construction in Section 10 would fail when we tried to create matrix M_2 from M_1.

The rest of the proof continues as before, with $XY = O(N^{1/2})$ and $W = \Theta(N^{3/4})$. (The fact that XY differs from $W^{2/3}$ by a constant multiple merely means that we have to do some trial and error.)

12. Extension to More Variables

Suppose we have a polynomial $p(x, y, z)$ in three variables over the integers. (The following remarks, suitably adapted, will also apply to a polynomial $p(x, y)$ in two variables modulo N.)

We could try to mimic the present approach. If the ranges X, Y, Z are small enough, we will end up with a polynomial relation $C(x, y, z)$, not a multiple of p, which is satisfied by (x_0, y_0, z_0). Then the resultant of $p(x, y, z)$ and $C(x, y, z)$ with respect to z will give a polynomial $r(x, y)$ in two variables. We can then try to solve $r(x, y) = 0$ by the current methods. But the degree of $r(x, y)$ will be quite high, so that the ranges X and Y which can be tolerated will be quite small.

A much more promising approach, which works often but not always, is as follows. If the ranges X, Y, Z are small enough, we are guaranteed to find a space of codimension 1 (a hyperplane) containing all the short vectors of the lattice $\hat{M}$. But we might easily find a space of larger codimension. (There is a good possibility that for many basis vectors $\mathbf{b}_i$ the orthogonal component $|\mathbf{b}_i^*|$ exceeds our known upper bound on $|\mathbf{s}|$, and each one increases the codimension of the space containing all the short vectors.) We develop several polynomial equations $C_i(x, y, z)$ satisfying $C_i(x_0, y_0, z_0) = 0$; the number of such equations is equal to the codimension of this space. We can then take resultants and g.c.d.s of the various $C_i(x, y, z)$ and $p(x, y, z)$ and hope to produce a single polynomial equation in a single variable $r(x) = 0$, which we solve over the reals.

This is only a heuristic approach, which might or might not work for a given polynomial p. One potential obstacle is that we might not obtain enough equations $C_i(x, y, z) = 0$. A related concern is that the equations we obtain might be redundant: for example, we might have $C_1(x, y, z) = xC_2(x, y, z)$. We see no way to guarantee that we will gather enough independent equations to enable a solution.

Indeed, certain counterexamples show that this procedure must fail for some polynomials. In the case $p(x, y) = 0 \pmod N$, we adapt an example of Manders and Adleman [10]. Let $n = q_1 q_2 \dots q_m$ be the product of the first m odd primes. Then $\log n \approx m \log m$. Also the modular equation $x^2 = 1 \pmod n$ has 2^m solutions. Let $N = n^h$ for a sufficiently large integer h. Look for solutions to

$$\begin{aligned} p(x, y) &= x^2 - yn - 1 = 0 \pmod N, \\ |x| &< n = X, \\ |y| &< n = Y. \end{aligned}$$

There are at least 2^{m+1} pairs (x_0, y_0) satisfying these equations. We cannot hope to produce them all in time polynomial in $\log N$. And yet we can arrange that any criterion

$$XY < N^{\varepsilon}$$

such as the hypothesis of Theorem 1 can be satisfied by proper choice of $h > 2/\varepsilon$.

If faced with this problem, our algorithm will probably return the equation

$$p(x, y) = x^2 - yn - 1 = 0$$

(moving it from a modular equation to an integer equation), but then be unable to proceed further (if we maintain $X = n$) because the appropriate bounds $XY < W^c$ or $X < n^c$ are not satisfied. If we allow $X = n^{1/2}$ it may still be able to proceed, but we will not have exponentially many solutions in this case.

However, we need not overemphasize the negative. The extended algorithm will often work, even in cases when it is not guaranteed.

An important application of the extended algorithm was alluded to in Section 7. Suppose a plaintext message m, consists of two unknown blocks of bits and a known piece, is subjected to RSA encryption with exponent 3. The message may be:

$m =$ "TODAY'S KEY IS swordfish AND THE PASSWORD IS joe."

We can view this as two unknowns: $x =$ "swordfish" and $y =$ "joe," and one known piece,

$B =$ "TODAY'S KEY IS ——— AND THE PASSWORD IS —."

We presume that we know the lengths of x and y, or can iterate over their possible values.

The plaintext message is

$$m = B + 2^k x + y,$$

the ciphertext is

$$c = m^3 \pmod N,$$

and the polynomial which we wish to solve is

$$p(x, y) = c - (B + 2^k x + y)^3 = 0 \pmod N,$$

with a solution (x_0, y_0) suitably bounded, and with c, k, B, N known and x, y unknown.

The polynomial $p(x, y)$ has total degree 3. We select a bound $3t$ on the degree of the monomials in our algorithm, so that we have monomials $x^g y^h$ with $g + h < 3t$. We introduce polynomial equations

$$q_{ijk}(x, y) = x^i y^j p(x, y)^k = 0 \pmod{N^k},$$

with $j \leq 2$, $k \geq 1$, and $i + j + 3k < 3t$.

The determinant of the related matrix has powers of N totaling

$$\binom{4}{2} + \binom{7}{2} + \cdots + \binom{3t-2}{2} = \frac{3t^2(t-1)}{2}.$$

The powers of $1/X$ come to

$$\binom{1}{2}+\binom{2}{2}+\binom{3}{2}+\cdots+\binom{3t}{2}=\binom{3t+1}{3}=\frac{27t^3-1}{6}$$

as do the powers of $1/Y$. So our requirement becomes

$$\begin{aligned}(XY)^{27t^3-1} &< N^{9t^3-9t^2},\\ (XY) &< N^{1/3-\varepsilon},\end{aligned}$$

with $\varepsilon \approx 1/(3t)$.

If this requirement is met we will get at least one equation $C(x, y) = 0$.

We tried an experiment on a scaled-down version of this. Rather than a cubic equation, we used a quadratic equation of the form

$$(B_0 + 2^k x + y)^2 = c \pmod N.$$

We had variables x_0, y_0 bounded by 2^{23}, and a modulus $N \approx 2^{150}$. We used monomials of total degree bounded by 5, so that there were 21 monomials and ten polynomial equations. The resulting requirement, $(XY)^{35} < N^{13}$, was met handily: $2^{1610} < 2^{1950}$. The matrix was represented as integers, and was scaled in such a way that the desired solution $\mathbf{s}$ had Euclidean length about 10^{38}. We ran basis reduction on the resulting 21×21 matrix. The results were much better than expected: For each $i = 2, 3, \ldots, 21$ we had $|\mathbf{b}_i^*| \approx 10^{41} > |\mathbf{s}|$, while $|\mathbf{b}_1| = |\mathbf{s}| \approx 10^{38}$, so that instead of confining the short vectors to a hyperplane the algorithm actually confined them to a one-dimensional subspace—we could just read off the answer $\mathbf{s}$. The computation time was disappointing though: the lattice basis reduction required 45 hours. Clearly much experimentation needs to be done yet with more optimized lattice basis reduction algorithms.

13. Conclusion and Open Problem

We have shown algorithms for finding solutions to univariate modular polynomial equations $p(x) = 0 \pmod N$, and bivariate integer polynomial equations $p(x, y) = 0$, as long as the solutions are suitably bounded with respect to N or to the coefficients of p, respectively.

We used the coefficients of p to build a lattice containing a short vector based on the unknown solution (x_0, y_0); this need not be the shortest vector. We then used a novel application of lattice basis reduction methods: rather than search for the shortest vector, we confine all relatively short vectors to a hyperplane. The equation of this hyperplane, when applied to our special short vector, gives a polynomial over the integers satisfied by (x_0, y_0), from which the solution follows.

We showed several applications to RSA with small encryption exponent, and to integer factorization with partial knowledge. We believe that other applications will arise. For example, Patarin [11] pointed out that the method of padding a message by repetition: (“Attack at dawn ... Attack at dawn ...”) amounts to multiplying a short message (x = “Attack at dawn ...”) by $2^k + 1$. If the message is short enough, and RSA with small exponent is used, the present techniques can derive the message again.

Joye and Quisquater [8] give many other cryptographic applications of the techniques presented here.

The present paper shows several potential exposures concerning RSA with small exponent. Specific implementations of RSA should be examined with regard to these exposures.

Conventional wisdom states that RSA should not be applied directly to messages, but rather that the messages should be randomized in some way prior to encryption, for example, by the methods of Bellare and Rogaway [1]. The results of the present paper give particular reinforcement to this wisdom in the case of small encrypting exponent.

The paper does not show any weaknesses in the RSA signature scheme with a small validating exponent.

Acknowledgments

The author gratefully acknowledges enlightening discussions with Andrew Odlyzko. Matt Franklin and Mike Reiter's Crypto '95 rump session paper and subsequent discussions were useful. Jacques Patarin was independently working on the idea of unknown offsets in the RSA setting. Barry Trager helped the experimental effort by coding up an implementation the Lovász basis reduction algorithm for Axiom. Suggestions from the anonymous referees greatly improved the presentation of the material.

Appendix 1. Another Solution for Multiple Encryptions

This material is related to the application in Section 8, but only tangentially to the main paper.

In Section 8 we had two encryptions of the same message with different random pads. If instead of two encryptions we have several, say $k+1$, then we can mount other attacks which might tolerate larger fields of random padding. We sketch here an attack which (heuristically) seems to tolerate random padding up to α times the length of N where

$$\alpha < \frac{k-2}{6k-3} < \frac{1}{6}.$$

Let the ciphertexts be

$$\begin{aligned} A_0 &= m^3 \pmod N, \\ A_i &= (m+r_i)^3 \pmod N, \\ c_i = A_i - A_0 &= 3m^2 r_i + 3m r_i^2 + r_i^3 \pmod N, \end{aligned}$$

so that we know A_0, A_i, c_i, and N, but not m or r_i. We assume the padding is small:

$$|r_i| < N^{\alpha}.$$

For indices $i < j < p$ define $d_{ij} = r_i r_j (r_i - r_j)$ and $e_{ijp} = -r_i r_j r_p (r_i - r_j)(r_j - r_p)(r_p - r_i)$. The $C(k,2) = \binom{k}{2}$ linearly independent quantities d_{ij} each satisfy $|d_{ij}| <$

$N^{3\alpha}$, and the $C(k, 3)$ linearly independent quantities e_{ijp} each satisfy $|e_{ijp}| < N^{6\alpha}$. One can check the following identity:

$$d_{ij}c_p + d_{jp}c_i - d_{ip}c_j = e_{ijp} \pmod N.$$

This suggests lattice basis reduction on the row basis of the following matrix. M is a square upper triangular integer matrix of dimension $(C(k, 2) + C(k, 3))$. Its upper-left $C(k, 2) \times C(k, 2)$ block is the identity times an integer approximation to $N^{3\alpha}$. Its lower-left $C(k, 3) \times C(k, 2)$ block is 0. Its lower-right $C(k, 3) \times C(k, 3)$ block is N times the identity. Its upper-right $C(k, 2) \times C(k, 3)$ block has rows indexed by pairs of indices $(i, j), i < j$, and columns indexed by triples of indices $(i, j, p), i < j < p$. Column (i, j, p) has three nonzero entries: c_p at row (i, j), c_i at row (j, p), and $-c_j$ at row (i, p).

Consider the integer row vector $\mathbf{r}$ whose first $C(k, 2)$ entries are d_{ij}, and whose last $C(k, 3)$ entries are the integers $(e_{ijp} - (d_{ij}c_p + d_{jp}c_i - d_{ip}c_j))/N$. The product $\mathbf{r}M = \mathbf{s}$ has left-hand elements $d_{ij}N^{3\alpha}$ and right-hand elements e_{ijp}; all its entries are bounded by $N^{6\alpha}$. We hope that lattice basis reduction will find this row.

The determinant of M is $N^{3\alpha C(k,2)+C(k,3)}$. This is larger than $(N^{6\alpha})^{C(k,2)+C(k,3)}$ because of our choice of α. So $\mathbf{s}$ is among the shorter elements of the lattice generated by the rows of M.

Contrary to the rest of this paper, we actually want to find $\mathbf{s}$, not just confine it to a hyperplane. The difficulty in finding $\mathbf{s}$ depends on its rank among the short elements. If $|r_i|$ are much smaller than N^{α}, then we can hope that $\mathbf{s}$ is the shortest lattice element, and that lattice basis reduction methods can recover it efficiently. We do not here supply efficiency estimates or probabilities of success; we treat this as a heuristic attack.

Assuming that we can actually find $\mathbf{s}$, we will be able to recover the values r_i by taking g.c.d. of elements of $\mathbf{r} = \mathbf{s}M^{-1}$:

$$\begin{aligned}\text{g.c.d.}\{d_{1,2}, d_{1,3}, \ldots, d_{1,k}\} &= \text{g.c.d.}\{r_1r_2(r_1 - r_2), r_1r_3(r_1 - r_3), \ldots, r_1r_k(r_1 - r_k)\}\\ &= r_1 \times \text{g.c.d.}\{r_2(r_1 - r_2), r_3(r_1 - r_3), \ldots, r_k(r_1 - r_k)\},\end{aligned}$$

and hopefully the latter g.c.d. will be small enough to discover by exhaustive search. Having found r_i, we can recover m by Franklin and Reiter's technique.

If we have 14 encryptions of the same message $(k = 13)$, then we can tolerate a random padding of about 150 bits in a 1024-bit RSA message.

Appendix 2. Nearly Orthogonal Toeplitz Columns

In this Appendix we give a proof of the technical result needed in Section 10: that several columns of the matrix M_4 are "nearly orthogonal." A modification of this proof would apply to any Toeplitz matrix.

Proof of Lemma 3. Let $W = |\mathbf{v}_{\gamma(a,b)}| = |\tilde{p}_{ab}|$ be the largest coefficient of $\tilde{p}$. Select indices (c, d) to maximize the quantity

$$8^{(c-a)^2+(d-b)^2}|\tilde{p}_{cd}|.$$

Select the rows

$$\gamma(c + i, d + j), \qquad 0 \le i, j < k,$$

of M_4 to create the desired submatrix $\tilde{M}$. Define an index function $\mu(i, j) = ki + j$. Then the matrix element $\tilde{M}_{\mu(g,h),\mu(i,j)}$ is the coefficient of $x^{c+g}y^{d+h}$ in $x^i y^j \tilde{p}(x, y)$, namely

$$\tilde{M}_{\mu(g,h),\mu(i,j)} = \tilde{p}_{g-i+c,h-j+d}.$$

Multiply the $\mu(g, h)$ row of $\tilde{M}$ by $8^{2(c-a)g+2(d-b)h}$, and multiply the $\mu(i, j)$ column by $8^{-2(c-a)i-2(d-b)j}$, to create a new matrix M' with the same determinant. Its typical element is

$$M'_{\mu(g,h),\mu(i,j)} = \tilde{p}_{g-i+c,h-j+d}8^{2(c-a)(g-i)+2(d-b)(h-j)}.$$

From maximality of (c, d) we find

$$|\tilde{p}_{g-i+c,h-j+d}|8^{(g-i+c-a)^2+(h-j+d-b)^2} \leq |\tilde{p}_{cd}|8^{(c-a)^2+(d-b)^2},$$

from which

$$|\tilde{p}_{g-i+c,h-j+d}|8^{2(g-i)(c-a)+2(h-j)(d-b)} \leq |\tilde{p}_{cd}|8^{-(g-i)^2-(h-j)^2}.$$

Thus each diagonal entry of M' is $\tilde{p}_{cd}$, and each off-diagonal entry is bounded by $|\tilde{p}_{cd}|8^{-(g-i)^2-(h-j)^2}$. This implies that M' is diagonally dominant, because the absolute values of the off-diagonal entries in its $\mu(i, j)$ row sum to at most

$$\begin{aligned}
|\tilde{p}_{cd}| \times \sum_{(g,h)\neq(i,j)} 8^{-(g-i)^2-(h-j)^2} \\
&= |\tilde{p}_{cd}| \times \sum_{(a,b)\neq(0,0)} 8^{-a^2-b^2} \\
&= |\tilde{p}_{cd}| \times \left[-1 + \sum_{(a,b)} 8^{-a^2-b^2}\right] \\
&= |\tilde{p}_{cd}| \times \left[-1 + \left(\sum_{a} 8^{-a^2}\right)^2\right] < \tfrac{3}{4}|\tilde{p}_{cd}|.
\end{aligned}$$

Each eigenvalue of M' is within $\frac{3}{4}|\tilde{p}_{cd}|$ of $\tilde{p}_{cd}$, and so exceeds $\frac{1}{4}|\tilde{p}_{cd}|$ in absolute value. By choice of (c, d) we know

$$\begin{aligned}
8^{(c-a)^2+(d-b)^2}|\tilde{p}_{cd}| &\geq 8^0|\tilde{p}_{ab}| = W, \\
|\tilde{p}_{cd}| &\geq 8^{-2\delta^2}W, \\
\det(M') &\geq (\tfrac{1}{4}8^{-2\delta^2}W)^{k^2} = W^{k^2}2^{-6k^2\delta^2-2k^2}.
\end{aligned}$$

For the second claim of the lemma: If the largest coefficient of $\tilde{p}$ is either $\tilde{p}_{00}$ or $\tilde{p}_{\delta\delta}$, set $(c, d) = (a, b)$ and notice that $\tilde{M}$ is a triangular matrix whose diagonal entries have absolute value W. If the largest coefficient is either $\tilde{p}_{0\delta}$ or $\tilde{p}_{\delta 0}$, redefine the indexing function as $\mu(i, j) = ki + (k - 1 - j)$ so that again $\tilde{M}$ is a triangular matrix whose diagonal entries have absolute value W. Similar results hold if (a, b) is any corner of the Newton polygon associated with $\tilde{p}$. □

References

[1] M. Bellare and P. Rogaway, Optimal asymmetric encryption, *Advances in Cryptology—EUROCRYPT* ’94 (A. De Santis, ed.), pp. 92–111, LNCS, 950, Springer-Verlag, Berlin, 1995.

[2] D. Coppersmith, Finding a small root of a univariate modular equation, *Advances in Cryptology—EUROCRYPT* ’96 (U. Maurer, ed.), pp. 155–165, LNCS, 1070, Springer-Verlag, Berlin, 1996.

[3] D. Coppersmith, Finding a small root of a bivariate integer equation; factoring with high bits known, *Advances in Cryptology—EUROCRYPT* ’96 (U. Maurer, ed.), pp. 178–189, LNCS, 1070, Springer-Verlag, Berlin, 1996.

[4] D. Coppersmith, Factoring with a hint, IBM Research Report RC 19905, January 16, 1995.

[5] D. Coppersmith, M. Franklin, J. Patarin, and M. Reiter, Low-exponent RSA with related messages, *Advances in Cryptology—EUROCRYPT* ’96 (U. Maurer, ed.), pp. 1–9, LNCS, 1070, Springer-Verlag, Berlin, 1996.

[6] M. Franklin and M. Reiter, A linear protocol failure for RSA with exponent three, Rump Session, Crypto ’95 (not in proceedings).

[7] J. Hastad. Solving simultaneous modular equations of low degree, *SIAM J. Comput*. **17** (1988), 336–341.

[8] M. Joye and J.-J. Quisquater, Protocol failures for RSA-like functions using Lucas sequences and elliptic curves, Presented at the Cambridge Workshop on Cryptographic Protocols, Cambridge, April 14–18, 1996.

[9] A. K. Lenstra, H. W. Lenstra, Jr., and L. Lovász, Factoring polynomials with rational coefficients, *Math. Ann*. **261** (1982), 515–534.

[10] K. Manders and L. Adleman, NP-complete decision problems for binary quadratics, *J. Comput. System Sci*. **16** (1978), 168–184.

[11] J. Patarin, Personal communication, 1995.

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

[13] R. L. Rivest and A. Shamir, Efficient factoring based on partial information, *Advances in Cryptology—EUROCRYPT* ’85, pp. 31–34, LNCS, 219, Springer-Verlag, Berlin, 1986.

[14] D. E. Knuth, *The Art of Computer Programming*, vol. 2, 2nd edn., Section 4.6.1, Addison-Wesley, Reading, Massachusetts, 1981.

[15] S. A. Vanstone and R. J. Zuccherato, Short RSA keys and their generation, *J. Cryptology* **8**(2) (1995), 101–114.