The Fermat factorization method revisited

Robert Erra[*] Christophe Grenier[†]

30th June 2009

Abstract

We consider the well known Fermat factorization method, we call the *Fermat factorization equation* the equation solved by it: $\mathcal{P}(x, y) = (x + 2R)^2 - y^2 - 4N = 0$; where $N = p\,q > 0$ is a RSA modulus with primes p and q supposed of equal length.

This equation is a bivariate integer polynomial equation and we propose to solve it directly using Coppersmith's methods for bivariate integer polynomials. As we use them as a black box, our proofs will be brief.

We show a first result: we can factor N in a polynomial time if $|p - q| < N^{5/18}$. Using the fact that the Newton polygon of $\mathcal{P}(x, y)$ is in fact a lower triangle we show a better result: we can indeed factor N in a polynomial time if $|p - q| < N^{1/3}$. We conclude with proposals for future works.

1 Introduction

Fermat, in a letter to Mersenne around 1643, exposed an algorithm to factor odd integers by writing them as a difference of two squares. Fermat was responding to a challenge proposed by Mersenne, he has presented his method for the number 2 027 651 281. For a composite integer $N > 0$, if a difference such as

$$4\,N = a^2 - b^2 = (a + b)(a - b) \tag{1}$$

can be found and neither factor equals one, then we get non-trivial factors of N.

To find a solution to the equation (1), Fermat proposes to try different values of a until $a^2 - N$ is a square, beginning with $a = \lceil \sqrt{N} \rceil$ and following the algorithm (1). He shows that 12 iterations are sufficiant to factor the number 2 027 651 281, which was an impressive result for this time.

[*] ESIEA, SI&S Lab, erra@esiea.fr

[†] ESIEA, SI&S Lab & University Rennes I, grenier.christophe@gmail.com

Algorithm 1 : **Fermat factorisation method**

```
**Input**: an odd composite integer $N > 0$ ;
**Output**: a non trivial factor of $N$ ;
**Begin**:
$a = \lceil \sqrt{N} \rceil$;
$b = \sqrt{a^2 - N}$;
**While NotInteger**(b) **Do**
    $a = a + 1$ ;
    $b = \sqrt{a^2 - N}$ ;
**Endofwhile**;
    **Return** $(a - b, a + b)$.
**End**.
```

The Fermat factorization method is an iterative, or linear, search. For an integer $N = p\,q$, de Weger's [4] has shown that the efficiency of the Fermat factorization method is governed by the ratio

$$\mathcal{O}(\frac{\Delta^2}{4n^{1/2}})$$

where $\Delta = |p - q|$ is the *prime difference*.

So, as it was pointed out by de Weger, if $\Delta = \mathcal{O}(N^{1/4})$ then the Fermat factorization method is quite trivial. The fact that a too small Δ makes RSA insecure is a fact known for a long time.

Coppersmith has proposed in 1996 two seminal methods, based on the LLL algorithm, one to find small roots of an *univariate polynomial modular* equation [2] and one to find small roots of a *bivariate polynomial integer* equation [1]. Since the equation (1) is a bivariate polynomial integer equation, our idea is to *solve directly* it using Coppersmith's [1] methods as a black box.

From now, we will call the following bivariate polynomial integer equation (and its variants):

$$x^2 - y^2 - 4N = 0; \tag{2}$$

the *Fermat factorization* equation.

The paper is organized as follows:

- in section 2 we present the Coppersmith results we will use;
- in section 3 we present a first result: we can factor in a polynomial time a RSA modulus $N = p\,q$ if $|p - q| < N^{5/18}$;
- we will finish the section 3 with a better result: we can factor in a polynomial time a RSA modulus $N = p\,q$ if $|p - q| < N^{1/3}$;

- we conclude by giving some ideas for future works.

As it was pointed out by May [7], known applications of Coppersmith's method for bivariate integer polynomials equations are not so numerous: there is the so called "factoring with high bits known" presented by Coppersmith [3] and the May's result [7]. So, to the best of the authors's knowledge this is a new result obtained by Coppersmith's method for bivariate integer polynomials and as we use them as a black box, our proofs will be brief.

2 Coppersmith's results

We will use the two following well known Coppersmith's theorems [1].

Theorem 1 (Coppersmith [1]). *Let $\mathcal{P}(x,y) \in \mathbb{Z}[x,y]$ be an irreducible polynomial of maximum degree δ in each variable separately. Let W be defined as $W = \|\mathcal{P}(xX, yY)\|_\infty$, the absolute value of the largest entry in the coefficients vector of $\mathcal{P}(x\,X, y\,Y)$ and let X, Y be bounds on the desired solution (x_0, y_0). If*

$$X\,Y \leq W^{\frac{2}{3\delta}} \tag{3}$$

then, in time polynomial in $\log W$ and 2^δ, we can find all integer solutions (x_0, y_0) with $\mathcal{P}(x_0, y_0) = 0$, $x_0 \leq X$, $y_0 \leq Y$.

With the same notations, if the polynomial is of total degree δ we have a better result.

Theorem 2 (Coppersmith [1]). *Let $\mathcal{P}(x,y) \in \mathbb{Z}[x,y]$ be an irredutible polynomial of total degree δ. Let X, $Y \in \mathbb{N}$ and define $W = \|\mathcal{P}(xX, yY)\|_\infty$. Then we can find all pairs $(x_0, y_0) \in \mathbb{Z}^2$ satisfying*

$$\mathcal{P}(x_0, y_0) = 0 \text{ with } |x_0| < X, |y_0| < Y$$

in time polynomial in $\log W$ and δ provided that

$$XY \leq W^{\frac{1}{\delta}} 2^{-\mathcal{O}(\delta)}.$$

3 Fermat revisited with Coppersmith's methods

A direct approach of Fermat method gives $\mathcal{P}(x,y) = x^2 - y^2 - N$; this bivariate polynomial cancels for $x = (p+q)/2$ and $y = (p-q)/2$. We can make a variable change $x' = x + R$ with $R = \lceil\sqrt{N}\rceil$ and normalize. We obtain the following bivariate integer polynomial:

$$\mathcal{P}(x,y) = (x + 2R)^2 - y^2 - 4N \tag{4}$$

whose roots are $x_0 = p + q - 2R$ and $y_0 = p - q$.

Since we are looking for a bound based on prime factor difference, we state as an upper bound of y_0

$$Y = N^{\alpha}.$$

Then we use the inequality proven by de Weger [4]

$$p + q - 2\sqrt{N} < \frac{(p-q)^2}{4\sqrt{N}}$$

So, since $R = \lceil\sqrt{N}\rceil > \sqrt{N}$ and $x_0 = p + q - 2R$, we obtain the upper bound of x_0

$$X = N^{2\alpha - 1/2}$$

We also notice that $W = \|f(xX, yY)\|_{\infty} = 4N$.

Blömer and May have presented a toolkit [6] to maximize the bounds giving different constructions rules for different shapes of the so called *Newton polygon* of a polynomial. Understanding the *shape* of a polynomial is important to obtain the best results as we will see.

First, we will begin by using the Rectangular form [6] applied with parameters $a = 2$ and $b = 2$ which includes our polynomial (see figure 1).

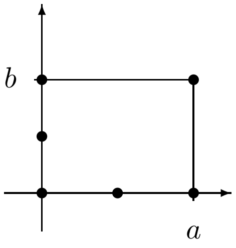

Figure 1: The Newton polygon of $ax^2 + by^2 + cx\,y + \cdots$ is a general rectangle

Using the first Coppersmith's theorem (1) and neglecting constants, applying the previous bounds and noticing that our polynomial is of degree 2 in each variable gives the inequality:

$$N^{2\alpha - 1/2} N^{\alpha} \leq N^{\frac{2}{6}} \tag{5}$$

that is equivalent to $\alpha \leq \frac{5}{18}$.

So, we have proved the following lemma.

Lemma 1. *Let $N = pq$ be an n-bit RSA modulus and p, q unknown primes, let $\Delta = |p - q| < N^{\alpha}$, then N can be factored in polynomial time if*

$$\alpha < \frac{5}{18}.$$

We conducted experiments (using *Mathematica*) for this bound and found acceptable results but we can obtain a better result if we use the fact that our polynomial has in fact no $(x\,y)$ term and so, it has a Newton polygon which is a *lower triangle* (figure (2)) and we can use now the theorem (2). Therefore, we apply our precedent upper bounds to obtain the inequality:

$$N^{2\alpha-1/2}N^{\alpha} \leq N^{\frac{1}{2}}$$

that is equivalent to:

$$\alpha \leq \frac{1}{3}$$

proving the following result:

Lemma 2. *Let $N = pq$ be an n-bit RSA modulus and p, q unknown primes. Let $\Delta = |p - q| < N^{\alpha}$. N can be factored in polynomial time if*

$$\alpha < \frac{1}{3}.$$

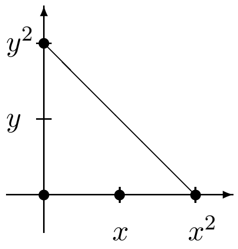

Figure 2: The Newton polygon of $ax^2 + bx + cy^2 + d$ is a lower triangle

This bound of 1/3 corresponds for a standard balanced RSA-1024 bits to factors p and q of 512 bits having their 171 most significant bits alike out of 512. For the bound of 5/18, the need is 228 most significant bits equal. It must be compared to the initial result of 256 bits so, the gain is, respectively, 85 and 28 bits.

4 Conclusion and future work

The last result we get confirms the idea widely exposed of the need of adequation between the Newton's polygon and the form to use in Coppersmith's method. We need to make more experiments to test these ideas but we think there are other interesting problems that can be studied with Coppersmith's methods. For example, we propose for future work to see what results we can have if:

- we look at $(x + 2R)^2 - y^2 = 0 \mod 4N$, the *modular Fermat factorization* equation ;

- we increase the number of variables, for example by considering the *trivariate* polynomial integer RSA equation $(x + y\,R)^2 - z^2 - 4N = 0$ or its modular form $(x + y\,R)^2 - z^2 = 0 \mod 4N$;

- we change the equation, considering for example the square root equation $(x + y\,R)^2 - 1 = 0 \mod N$ which is a modular bivariate equation or the trivariate integer polynomial equation $(x + y\,R)^2 - 1 - zN = 0$;

- we combine the method of ”factoring with high bits known” [3] with the approach presented here ?

It is not so simple because in almost all these cases it is well known that Coppersmith’s methods are usually only heuristic but, as usual in cryptanalysis, a cryptanalytic method is always interesting even in the case it is not fully general. We thank Éric Filiol to have convinced us to put down these results, that are a part of a longer paper [5], and Vincent Guyot for improving the presentation.

References

[1] D. Coppersmith. Finding a small root of a bivariate integer equation; factoring with high bits known. *In Proceedings of Eurocrypt’96, Lecture Notes in Computer Science*, 1996.

[2] D. Coppersmith. Finding a small root of a univariate modular equation. *In Proceedings of Eurocrypt’96, Lecture Notes in Computer Science*, pages 155–165, 1996.

[3] D. Coppersmith. Small solutions to polynomial equations, and low exponent RSA vulnerabilities. *Journal of Cryptology*, 10:233–260, 1997.

[4] B. de Weger. Cryptanalysis of RSA with small prime difference. *AAECC*, 13(1):17–28, 2002.

[5] R. Erra and C. Grenier. How to choose a RSA key ? submitted to the *i*AWACS’09 Workshop in LAVAL/FRANCE 2009.

[6] E. Jochemz and 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, 2006.

[7] A. May. Computing the RSA Secret Key is Deterministic Polynomial Time Equivalent to Factoring. *In Proceedings of Crypto’04, volume 3152 of Lecture Notes in Computer Science*, pages 213–219, 2004.