

## Re: I was right, surrogate factoring proof

**Source:** <http://sci.tech-archive.net/Archive/sci.math/2005-02/4765.html>

---

**From:** Décio Luiz Gazzoni Filho (*decio\_at\_decpp.removethis.net*)

**Date:** 02/13/05

Date: Sun, 13 Feb 2005 16:43:53 -0200

jstevh@msn.com wrote:

> *Here is the proof.*

>

> *Given*

>

>  $yx^2 + Ax - M^2 = 0$

>

>  $yz^2 + Az - j^2 = 0$

>

> *and*

>

>  $T = M^2 - j^2$

>

> *it follows that*

>

>  $x = z(-Az \pm \sqrt{(Az - 2M^2)^2 - 4TM^2}) / (2j^2 - 2Az)$

>

> *and*

>

>  $z = x(-Ax \pm \sqrt{(Ax - 2j^2)^2 + 4Tj^2}) / (2M^2 - 2Ax)$

>

> *and, you can multiply both equations by A, to*

>

> *get*

>

>  $Ax = Az(-Az \pm \sqrt{(Az - 2M^2)^2 - 4TM^2}) / (2j^2 - 2Az)$

>

>  $Az = Ax(-Ax \pm \sqrt{(Ax - 2j^2)^2 + 4Tj^2}) / (2M^2 - 2Ax)$

>

> *where for any rational x and z there must be an integer A such that*

> *both Ax and Az are integers, so looping through \*all\* possible integer*

> *Ax and Ay as required by the square roots MUST factor M.*

>

> *So, I can't be wrong, or, algebra is wrong.*

>

> *If you disagree, show a false step.*

James,

I'm going to compile Tim Peters' and Rick Decker's arguments in an easy-to-follow message and include a lot of concrete examples. I hope you don't disregard this reply, because it points the fatal flaw that you've been ignoring up until now.

To start off, I'm not arguing the validity of your method. I'm not checking it either, so you may have made a mistake or two in the signs (and then again, you may not). For now let's just assume that it works. So basically, you've reduced the problem of finding a factor of  $M$  to the problem of finding a suitable  $A$ .

Now I'll introduce you to a related idea. Or maybe you know it already. Suppose that I have a pair of integers  $x$  and  $y$ , such that  $x \not\equiv y \pmod{n}$  and  $x \not\equiv -y \pmod{n}$ , but  $x^2 \equiv y^2 \pmod{n}$ . Now  $\gcd(x-y, n)$  is a nontrivial factor of  $n$ . See how I've reduced the problem of finding a factor of  $M$  to finding two suitable values of  $x$  and  $y$ ?

However, there's a problem. Assuming that  $n = p \cdot q$  is the product of two prime factors, then for a given  $x$ , there are only 4 values of  $y$  such that  $x^2 \equiv y^2 \pmod{n}$ , and of those 4 values, two of these are  $x$  and  $-x$ , so they're not valid. So, unless we have an efficient method to search for the 2 valid values, we'll have to search through a sizeable amount of the interval  $[1, n]$  before finding an answer. Now you have to agree that I might just as well generate primes from 1 to  $\sqrt{n}$  and test whether these primes divide  $n$  — at most I'll have to look through  $\sqrt{n}/(2 \log n)$  primes, which is much better than searching a range the size of  $n$ .

There's a very important difference between your idea and this idea that I just described, which is known as congruence of squares. For the latter, some very bright researchers have devised efficient ways of sieving through values of  $x$  and  $y$  to find suitable values. You may have heard of these methods — they're called the Quadratic Sieve and the Number Field Sieve.

You think that you've solved the factoring problem just because one can, in principle, search through all possible values of  $A$  until finding a suitable one. But don't forget that's included in the cost of the algorithm! One doesn't measure the cost of QS or NFS just as the time to compute  $\gcd(x-y, n)$ , but rather as the whole cost of finding these suitable values of  $x$  and  $y$  and then taking the gcd. So, if surrogate factoring ever becomes a world-renowned and respected factoring algorithm, it won't be because of the algebra you did above, but because of an efficient method that you provide to find the suitable  $A$  that splits  $M$ .

QS and NFS aren't the only examples of this. The ECM method also gives an easy way to factor  $n$ , as long as you find a suitable elliptic curve group (which is to say, one of smooth order.) Now the big deal about ECM is that some bright researchers have analyzed it and guaranteed us that by generating elliptic curves at random, we'll somewhat quickly find a suitable one — not as quickly as we desired (i.e. in polynomial time), but quicker than previous methods, and that's why ECM is so valuable.

Pollard rho works the same way. The basic idea is that if you find two distinct values  $x$  and  $y$  (distinct modulo  $n$ , that is), which lie in the same position in a pseudorandom cycle modulo one of the factors of  $n$ , then these two values are congruent modulo  $p$ , and hopefully not congruent modulo  $n/p$ . Thus  $\gcd(x-y, n)$  should reveal a non-trivial factor of  $n$ . This by itself isn't very useful, but Pollard suggested a somewhat efficient way to seek values of  $x$  and  $y$ : use a pseudorandom function  $f(x)$  (usually  $f(x) = x^2 + 1$ ), then iterate through the two sequences  $f(x)$  and  $f(f(x))$ . Eventually those two sequences will collide, and using either Floyd's or Bren't cycle-finding algorithms one can easily spot the collision between these two sequences.

So, is it clear to you that your method is useless, unless you provide an efficient way to find the required parameters? I have shown that one can efficiently find  $j$  such that  $M + j$  and  $M - j$  are trivially factorable. Can you efficiently find  $A$ ? I suggest you concentrate your research on this, because that's the big gap in your algorithm.

Décio