

Re: JSH: Surrogate factoring, periodic behavior

Source: <http://sci.tech--archive.net/Archive/sci.math/2007-09/msg00082.html>

- *From:* rossum <rossum48@xxxxxxxxxxxxx>
 - *Date:* Sat, 01 Sep 2007 11:34:29 +0100
-

On Sat, 01 Sep 2007 05:21:41 -0000, JSH <jstevh@xxxxxxxxxx> wrote:

Factored trying 5 surrogates on the fifth surrogate using $k=1$ where the start was with $n=7$.

I ran my usual tests again on 500 random composite odd numbers that are multiples of two different primes, each in the range 500 to 1000. The results are compared to Fermat's method, trial factorisation (both forward and reverse) and random picking.

Fermat average = 7.58 probes.
JSH average = 1635.83 probes.
Probe ratio = 1 : 215.752
Trial average = 118.52 probes.
Reverse average = 12.12 probes.
Random average = 727.79 probes.

500 trials, 0 misfactors found.

Average k 's tried per factorisation: 1.000
Average n 's tried per factorisation: 47.040

k was fixed at 1 and n was 7, 8, 9, ...

Part of the problem I've run into with comments about surrogate factoring have been repeated claims it works only as good as trial division, but hey, I programmed the damn thing.

Any chance of seeing your code?

Mine is below. It is Java. This is not compilable as-is because there are a few external classes and functions that I use. They are all pretty obvious.

```
// -- Begin Code --
```

Re: JSH: Surrogate factoring, periodic behavior

```
// JSH method – August 07
static FactorData jshFactor(int target) {
    FactorData retData = new FactorData();
    retData.factor1 = 0;
    retData.factor2 = 0;
    retData.probeCount = 0;

    int k = 1;
    int n = 7;

    while(true) {
        // Surrogate to factor = 2k^2 + nT
        long surrogate = Math.abs(2L * k * k + n * target);

        ArrayList<Ipair> sFactors = allFactorPairs(surrogate);

        ++nCount;

        // Try possible solutions
        long g_1, g_2, trialFac;
        for (int i = 0; i < sFactors.size(); ++i) {
            g_1 = sFactors.get(i).first;
            g_2 = sFactors.get(i).second;

            // fac1, positive
            trialFac = Maths.GCD(g_1 - k, target);
            ++retData.probeCount;
            if (trialFac > 1 && trialFac < target) {
```