Revisiting the Binary Euclidean Algorithm*

Richard P. Brent
Computing Laboratory
University of Oxford

rpb@comlab.ox.ac.uk

27 Sept 2001

*Copyright ©2001, R. P. Brent. HK3t

Abstract

The binary Euclidean algorithm is a variant of the classical Euclidean algorithm. It avoids divisions and multiplications, except by powers of two, so is potentially faster than the classical algorithm on a binary machine.

I will describe the binary algorithm and consider its average case behaviour. In particular, I will correct some small but interesting errors in the literature, discuss some recent results of Brigitte Vallée, and describe a numerical computation which verifies a conjecture of Vallée to 40 decimal places.

Outline

- The binary Euclidean algorithm
  - First formulation (Algorithm B)
  - History
  - Worst case, extensions etc
- Continuous model
  - Conjectured/empirical results
- Another formulation (Algorithm V)
  - Connection with continued fractions
  - Vallée's average-case results
- Useful operators
  - Relation between the operators
  - Fixed points
  - Vallée's conjectures and results
- Numerical results
- Tracking down an error

Notation

$\lg(x)$ denotes $\log_2(x)$.

N, n, u, v are positive integers.

$\mathrm{Val}_2(u)$ denotes the dyadic valuation of the positive integer u, i.e. the greatest integer j such that $2^j \mid u$.

The Binary Euclidean Algorithm

The idea of the *binary* Euclidean algorithm is to avoid the "division" operation $r \leftarrow m \bmod n$ of the classical algorithm, but retain $O(\log N)$ worst (and average) case.

We assume that the algorithm is implemented on a binary computer so division by a power of two is easy. In particular, we assume that the "shift right until odd" operation

$$u \leftarrow u/2^{\mathrm{Val}_2(u)}$$

or equivalently

$$\text{while even}(u) \text{ do } u \leftarrow u/2$$

can be performed in constant time, although time $O(\mathrm{Val}_2(u))$ would be sufficient.

Definition of the Binary Euclidean Algorithm

There are several almost equivalent ways to define the algorithm. It is easy to take account of the largest power of two dividing the inputs, so for simplicity we assume that u and v are *odd* positive integers.

Following is a simplified version of the algorithm given in Knuth, §4.5.2.

Algorithm B

B1. $t \leftarrow |u - v|$;
if $t = 0$ terminate with result u

B2. $t \leftarrow t/2^{\mathrm{Val}_2(t)}$

B3. if $u \geq v$ then $u \leftarrow t$ else $v \leftarrow t$;
go to B1.

History

The binary Euclidean algorithm is attributed to Silver and Terzian (unpublished, 1962) and Stein (1967). However, it seems to go back almost as far as the classical Euclidean algorithm. Knuth (§4.5.2) quotes a translation of a first-century AD Chinese text *Chiu Chang Suan Shu* on how to reduce a fraction to lowest terms:

> If halving is possible, take half.
>
> Otherwise write down the denominator and the numerator, and subtract the smaller from the greater.
>
> Repeat until both numbers are equal.
>
> Simplify with this common value.

This looks very much like Algorithm B !

The Worst Case

At step B1, u and v are odd, so t is even. Thus, step B2 always reduces t by at least a factor of two. Using this fact, it is easy to show that step B3 is executed at most

$$\lfloor \lg(u + v) \rfloor$$

times (Knuth, exercise 4.5.2.37). Thus, if $N = \max(u, v)$, step B3 is executed at most

$$\lg(N) + O(1)$$

times.

Remark

Even if step B2 is replaced by single-bit shifts

$$\text{while even}(t) \text{ do } t \leftarrow t/2$$

the overall worst case time is still $O(\log N)$.

Extended Binary Algorithm

It is possible to give an *extended* binary GCD algorithm which computes multipliers α and β such that

$$\alpha u + \beta v = \mathrm{GCD}(u, v)$$

(Bojanczyk and Brent, 1987).

Parallel GCD Algorithms

There is a systolic array variant of the binary GCD algorithm (Brent and Kung, 1985). This takes time $O(\log N)$ using $O(\log N)$ 1-bit processors. The overall bit-complexity is $O(\log N)^2$.

For n-bit numbers the systolic algorithm gives time $O(n)$ using $O(n)$ processors. This is close to the best known parallel time bound. It is not known if GCD is in the class **NC** (polylog time with a polynomial number of processors).

A Heuristic Continuous Model

To analyse the expected behaviour of Algorithm B, we can follow what Gauss did for the classical algorithm. This was first attempted in my 1976 paper[1] and there is a summary in Knuth (Vol. 2, *third* edition, §4.5.2).

Assume that the initial inputs u_0, v_0 to Algorithm B are uniformly and independently distributed in (0, N), apart from the restriction that they are odd. Let (u_n, v_n) be the value of (u, v) after n iterations of step B3.

Let

$$x_n = \frac{\min(u_n, v_n)}{\max(u_n, v_n)}$$

and let $F_n(x)$ be the probability distribution function of x_n (in the limit as $N \to \infty$). Thus $F_0(x) = x$ for $x \in [0, 1]$.

[1] R. P. Brent, Analysis of the Binary Euclidean Algorithm, *New Directions and Recent Results in Algorithms and Complexity*, (J. F. Traub, editor), Academic Press, New York, 1976, 321–355.

Plausible Assumption

We make the plausible (but not proved) assumption[2] that $\mathrm{Val}_2(t)$ takes the value k with probability 2^{-k} at step B2.

It is plausible because $\mathrm{Val}_2(t)$ at step B2 depends on the least significant bits of u and v, whereas the comparison at step B3 depends on the most significant bits, so one would expect the steps to be (almost) independent.

[2] Vallée does not need to make this assumption. Her results will be mentioned later.

The Recurrence for F_n

Consider the effect of steps B2 and B3. We can assume that $u > v$ so $t = u - v$. If $\mathrm{Val}_2(t) = k$ then $X = v/u$ is transformed to

$$\begin{aligned} X' &= \min\left(\frac{u-v}{2^k v}, \frac{2^k v}{u-v}\right) \\ &= \min\left(\frac{1-X}{2^k X}, \frac{2^k X}{1-X}\right) . \end{aligned}$$

It follows that $X' < x$ iff

$$X < \frac{1}{1+2^k/x} \text{ or } X > \frac{1}{1+2^k x} .$$

Thus, the recurrence for $\widetilde{F}_n(x) = 1 - F_n(x)$ is

$$\widetilde{F}_{n+1}(x) = \sum_{k\ge 1} 2^{-k} \left(\widetilde{F}_n\left(\frac{1}{1+2^k/x}\right) - \widetilde{F}_n\left(\frac{1}{1+2^k x}\right)\right)$$

and $\widetilde{F}_0(x) = 1 - x$ for $x \in [0, 1]$.

The Recurrence for f_n

Differentiating the recurrence for $\widetilde{F}_n$ we obtain (formally) a recurrence for the probability density $f_n(x) = F_n'(x) = -\widetilde{F}_n'(x)$:

$$\begin{aligned} f_{n+1}(x) &= \sum_{k\ge1} \left(\frac{1}{x+2^k}\right)^2 f_n\left(\frac{x}{x+2^k}\right) \\ &+ \sum_{k\ge1} \left(\frac{1}{1+2^k x}\right)^2 f_n\left(\frac{1}{1+2^k x}\right) . \end{aligned}$$

Operator Notation

The recurrence for f_n may be written as

$$f_{n+1} = \mathcal{B}_2 f_n,$$

where the operator $\mathcal{B}_2$ is the case $s = 2$ of a more general operator $\mathcal{B}_s$ which will be defined later.

Conjectured and Empirical Results

In my 1976 paper I gave numerical and analytic evidence that $F_n(x)$ converges to a limiting distribution $F(x)$ as $n \to \infty$, and that $f_n(x)$ converges to the corresponding probability density $f(x) = F'(x)$ (note that $f = \mathcal{B}_2 f$ so f is a "fixed point" of the operator $\mathcal{B}_2$).

Assuming the existence of F, it is shown in my 1976 paper that the expected number of iterations of Algorithm B is $\sim K \lg N$ as $N \to \infty$, where $K = 0.705\ldots$ is a constant defined by

$$K = \ln 2 / E_\infty ,$$

and

$$E_\infty = \ln 2 + \int_0^1 \left(\sum_{k=2}^{\infty} \left(\frac{1-2^{-k}}{1+(2^k-1)x} \right) - \frac{1}{2(1+x)} \right) F(x)\, dx .$$

A Simplification

We can simplify the expression for K to obtain

$$K = 2/b ,$$

where

$$b = 2 - \int_0^1 \lg(1-x) f(x)\, dx .$$

Using integration by parts we obtain an equivalent expression

$$b = 2 + \frac{1}{\ln 2} \int_0^1 \frac{1-F(x)}{1-x}\, dx .$$

For the proofs, see Knuth, third edition, §4.5.2.

Another Formulation – Algorithm V

It will be useful to rewrite Algorithm B in the following equivalent form (using pseudo-Pascal):

Algorithm V { Assume $u \le v$ }

while $u \ne v$ **do**
  begin
  while $u < v$ **do**
    begin
    $j \leftarrow \mathrm{Val}_2(v-u)$;
    $v \leftarrow (v-u)/2^j$;
    end;
  $u \leftrightarrow v$;
  end;
return u.

Continued Fractions

Vallée [15] shows a connection between Algorithm V and continued fractions of a certain form:

$$\frac{u}{v} = 1/a_1 + 2^{k_1}/a_2 + 2^{k_2}/\ldots/a_r + 2^{k_r} ,$$

where a_j is odd, $k_j > 0$, and $0 < a_j < 2^{k_j}$.

Example

Consider $u = 9$, $v = 55$. The inner loop of Algorithm V finds

$$\begin{aligned} 55 &= 9 + 2 \times 23 \\ &= 9 + 2 \times (9 + 2 \times 7) \\ &= 3 \times 9 + 4 \times 7 \end{aligned}$$

and on the next iteration

$$9 = 7 + 2 \times 1$$

so

$$\frac{55}{9} = 3 + 4 \times \frac{7}{9} ,$$

$$\frac{9}{7} = 1 + \frac{2}{7} ,$$

and finally

$$\frac{9}{55} = \frac{1}{3 + \frac{4}{1 + \frac{2}{3+4}}}$$

which by convention we write as

$$\frac{9}{55} = 1/3 + 4/1 + 2/3 + 4 .$$

Vallée's Results – More Details

Algorithm V has two nested loops. The outer loop exchanges u and v. Between two exchanges, the inner loop performs a sequence of subtractions and shifts which can be written as

$$\begin{aligned} v &\to u + 2^{b_1} v_1; \\ v_1 &\to u + 2^{b_2} v_2; \\ &\cdots \\ v_{m-1} &\to u + 2^{b_m} v_m \end{aligned}$$

with $v_m < u$.

If $x_0 = u/v$ at the beginning of an inner loop, the effect of the inner loop followed by an exchange is the rational $x_1 = v_m/u$ defined by

$$x_0 = \frac{1}{a + 2^k x_1} ,$$

where a is an odd integer given by

$$a = 1 + 2^{b_1} + 2^{b_1+b_2} + \cdots + 2^{b_1+\cdots b_{m-1}} ,$$

and the exponent k is given by

$$k = b_1 + \cdots b_m .$$

Thus, the rational u/v, for $1 \le u < v$, has a unique *binary continued fraction expansion* of the form

$$\frac{u}{v} = \cfrac{1}{a_1 + \cfrac{2^{k_1}}{a_2 + \cfrac{2^{k_2}}{\ddots \; + \cfrac{2^{k_{r-1}}}{a_r + 2^{k_r}}}}}$$

Vallée studies three parameters related to this continued fraction

1. The height or the depth (i.e. the number of exchanges) r.
2. The total number of operations necessary to obtain the expansion; if $p(a)$ denotes the number of "1"s in the binary expansion of the integer a, it is equal to $p(a_1) + p(a_2) + \cdots + p(a_r)$.
3. The sum of exponents of 2 in the numerators of the binary continued fraction, $k_1 + \cdots + k_r$.

Vallée's Theorems

Vallée's main results give the average values of the three parameters above: the average values are asymptotically $A_i \log N$ for certain constants A_1, A_2, A_3 related to the spectral properties of an operator $\mathcal{V}_2$ (to be defined later).

Some Useful Operators

Operators $\mathcal{B}_s$, $\mathcal{U}_s$, $\widetilde{\mathcal{U}}_s$, $\mathcal{V}_s$, useful in the analysis of the binary Euclidean algorithm, are defined on suitable function spaces by

$$\mathcal{U}_s[f](x) = \sum_{k\geq 1} \left(\frac{1}{1+2^k x}\right)^s f\left(\frac{1}{1+2^k x}\right), \quad (1)$$

$$\widetilde{\mathcal{U}}_s[f](x) = \left(\frac{1}{x}\right)^s \mathcal{U}_s[f]\left(\frac{1}{x}\right), \quad (2)$$

$$\mathcal{B}_s = \mathcal{U}_s + \widetilde{\mathcal{U}}_s,$$

$$\mathcal{V}_s[f](x) = \sum_{k\geq 1} \sum_{\substack{a \text{ odd},\\ 0<a<2^k}} \left(\frac{1}{a+2^k x}\right)^s f\left(\frac{1}{a+2^k x}\right). \quad (3)$$

In these definitions s is a complex variable, and the operators are called Ruelle operators. They are linear operators acting on certain function spaces.

The case $s = 2$ is of particular interest. $\mathcal{B}_2$ encodes the effect of one iteration of the inner "while" loop of Algorithm V, and $\mathcal{V}_2$ encodes the effect of one iteration of the outer "while" loop.

History and Notation

$\mathcal{B}_2$ (denoted T) was introduced in my 1976 paper and was generalised to $\mathcal{B}_s$ by Vallée. $\mathcal{V}_s$ was introduced by Vallée. We shall call

- $\mathcal{B}_2$ the *binary Euclidean operator* and
- $\mathcal{V}_s$ *Vallée's operator.*

Relation Between the Operators

The operators are closely related, as the following Lemma and Theorem show.

Lemma 1

$$\mathcal{V}_s = \mathcal{V}_s \widetilde{\mathcal{U}}_s + \mathcal{U}_s.$$

Since I should prove *something* today, I will give an algebraic proof of Lemma 1.

Proof of Lemma 1

From (3),

$$\mathcal{V}_s[\widetilde{\mathcal{U}}_s[f]](x) = \sum_{k\geq 1} \sum_{\substack{a \text{ odd},\\ 0<a<2^k}} \left(\frac{1}{a+2^k x}\right)^s \widetilde{\mathcal{U}}_s[f]\left(\frac{1}{a+2^k x}\right)$$

but, from (1) and (2),

$$\widetilde{\mathcal{U}}_s[f](y) = \sum_{m\geq 1} \left(\frac{1}{2^m+y}\right)^s f\left(\frac{1}{1+2^m/y}\right).$$

On substituting $y = 1/(a+2^k x)$ we obtain

$$\mathcal{V}_s[\widetilde{\mathcal{U}}_s[f]](x) = \sum_{k\geq 1} \sum_{\substack{a \text{ odd},\\ 0<a<2^k}} \sum_{m\geq 1} \left(\frac{1}{1+2^m a+2^{k+m}x}\right)^s \times f\left(\frac{1}{1+2^m a+2^{k+m}x}\right).$$

Proof of Lemma 1 cont.

Thus, to show that

$$\mathcal{V}_s[f](x) = \mathcal{U}_s[f](x) + \mathcal{V}_s\widetilde{\mathcal{U}}_s[f](x)$$

it suffices to observe that the set of polynomials

$$\{a' + 2^{k'}x \mid k' \geq 1, a' \text{ odd}, 0 < a' < 2^{k'}\}$$

is the disjoint union of the two sets

$$\{1 + 2^k x \mid k \geq 1\}$$

and

$$\{1 + 2^m a + 2^{k+m}x \mid k \geq 1, m \geq 1, a \text{ odd}, 0 < a < 2^k\}.$$

To see this, consider the two cases $a' = 1$ and $a' > 1$. If $2^{k'} > a' > 1$ we can write $a' = 1 + 2^m a$, $k' = k + m$, for some (unique) odd a and positive k, m. □

Algorithmic interpretation

Algorithm V gives an interpretation of Lemma 1 in the case $s = 2$. If the input density of $x = u/v$ is $f(x)$ then execution of the inner "while" loop followed by the exchange of u and v transforms this density to $\mathcal{V}_2[f](x)$. However, by considering the first iteration of this loop (followed by the exchange if the loop terminates) we see that the transformed density is given by

$$\mathcal{V}_2\widetilde{\mathcal{U}}_2[f](x) + \mathcal{U}_2[f](x),$$

where the first term arises if $u < v$ without an exchange, and the second arises if an exchange occurs.

Consequence of Lemma 1

The following Theorem gives a simple relationship between $\mathcal{B}_s$, $\mathcal{V}_s$ and $\mathcal{U}_s$.

Theorem 1

$$(\mathcal{V}_s - \mathcal{I})\mathcal{U}_s = \mathcal{V}_s(\mathcal{B}_s - \mathcal{I}) \ .$$

Proof. This is immediate from Lemma 1 and the definitions of the operators. □

Fixed Points

It follows immediately from Theorem 1 that, if

$$g = \mathcal{U}_2 f,$$

then

$$(\mathcal{V}_2 - \mathcal{I})g = \mathcal{V}_2(\mathcal{B}_2 - \mathcal{I})f.$$

Thus, if f is a fixed point of the operator $\mathcal{B}_2$, then g is a fixed point of the operator $\mathcal{V}_2$. From the recent result of Vallée we know that $\mathcal{V}_2$, acting on a certain Hardy space $\mathcal{H}^2(\mathcal{D})$, has a unique positive dominant simple eigenvalue 1, so g must be (a constant multiple of) the corresponding eigenfunction (provided $g \in \mathcal{H}^2(\mathcal{D})$). Also, from the definitions of $\mathcal{B}_2$ and $\mathcal{U}_2$, we have

$$\lambda = f(1) = 2g(1) = 2\sum_{k\geq 1}\left(\frac{1}{1+2^k}\right)^2 f\left(\frac{1}{1+2^k}\right),$$

which is useful for proving the consistency of two of the expressions for K given below.

Some Recent Results of Vallée

Using her operator $\mathcal{V}_s$, Vallée recently proved that

$$K = \frac{2\ln 2}{\pi^2 g(1)} \sum_{\substack{a \text{ odd}, \\ a>0}} 2^{-\lfloor \lg a \rfloor} G\left(\frac{1}{a}\right)$$

where g is a nonzero fixed point of $\mathcal{V}_2$ (i.e. $g = \mathcal{V}_2 g \neq 0$) and $G(x) = \int_0^x g(t)\,dt$. This is the only expression for K which has been rigorously proved.

Because $\mathcal{V}_s$ can be proved to have nice spectral properties, the existence and uniqueness (up to scaling) of g can be proved rigorously.

Warning: Knuth uses $G(x)$ for our $\tilde{F}(x) = 1 - F(x)$! Unfortunately Knuth and Vallée use incompatible notation. We have followed Vallée, with the exception that we do not assume the normalisation $G(1) = 1$.

A Conjecture of Vallée

Let $\lambda = f(1)$, where f is the limiting probability density (conjectured to exist) as above. Vallée (see Knuth, third edition, §4.5.2(61)) conjectured that

$$\frac{\lambda}{b} = \frac{2\ln 2}{\pi^2} ,$$

or equivalently that

$$K = \frac{4\ln 2}{\pi^2 \lambda} . \tag{4}$$

Vallée proved the conjecture under the assumption that the operator $\mathcal{B}_s$ satisfies a certain spectral condition.

Numerical Results

Using an improvement of the "discretization method" of my 1976 paper, and the MP package with the equivalent of more than 50 decimal places (50D) working precision, we computed the limiting probability density f, then K, $\lambda = f(1)$, and $K\lambda$. The results were

$$\begin{aligned} K &= 0.70597\,12461\,01916\,39152\,93141\,35852\,88176\,66677 \\ \lambda &= 0.39792\,26881\,18831\,66440\,76707\,16114\,26549\,82309\,8 \\ K\lambda &= 0.28092\,19710\,90731\,50563\,57543\,97987\,98803\,85315 \end{aligned}$$

These are believed to be correctly rounded values.

Vallée's conjecture (4) is that

$$K\lambda = 4\ln 2/\pi^2 .$$

The computed value of $K\lambda$ agrees with $4\ln 2/\pi^2$ to 40 decimals (in fact to 44 decimals)!

Correcting an Error

In my 1976 paper I claimed that, for all $n \geq 0$ and $x \in (0, 1]$,

$$F_n(x) = \alpha_n(x)\lg(x) + \beta_n(x) , \tag{5}$$

where $\alpha_n(x)$ and $\beta_n(x)$ are analytic and regular in the disk $|x| < 1$. However, *this is incorrect*, even in the case $n = 1$.

The error appeared to go unnoticed until 1997, when Don Knuth was revising Volume 2 in preparation for publication of the third edition. Knuth computed the constant K using recurrences for the analytic functions $\alpha_n(x)$ and $\beta_n(x)$, and I computed K directly using the defining integral and recurrences for $F_n(x)$. Our computations disagreed in the 14th decimal place ! Knuth found

$$K = 0.70597\ 12461\ 01945\ 99986\cdots$$

but I found

$$K = 0.70597\ 12461\ 01916\ 39152\cdots$$

Some Detective Work

After a flurry of emails we tracked down the error. It was found independently, and at the same time (within 24 hours), by Flajolet and Vallée.

The source of the error is illustrated by Lemma 3.1 of my 1976 paper, which is wrong (and corrected in the solution to ex. 4.5.2.29 of Knuth, third edition).

The *Mellin transform* of a function $g(x)$ is defined by

$$g^*(s) = \int_0^\infty g(x) x^{s-1} dx \; .$$

It is easy to see that, if

$$f(x) = \sum_{k\geq 1} 2^{-k} g(2^k x) \; ,$$

then the Mellin transform of f is

$$f^*(s) = \sum_{k\geq 1} 2^{-k(s+1)} g^*(s) = \frac{g^*(s)}{2^{s+1}-1} \; .$$

Mellin Inversion

Under suitable conditions we can apply the Mellin inversion formula to obtain

$$f(x) = \frac{1}{2\pi i} \int_{c-i\infty}^{c+i\infty} f^*(s) x^{-s} ds \; .$$

Applying these results to $g(x) = 1/(1+x)$, whose Mellin transform is $g^*(s) = \pi/\sin \pi s$ when $0 < \mathcal{R}s < 1$, we find

$$f(x) = \sum_{k\geq 1} \frac{2^{-k}}{1+2^k x}$$

as a sum of residues of

$$\left(\frac{\pi}{\sin \pi s}\right) \frac{x^{-s}}{2^{s+1}-1}$$

for $\mathcal{R}s \leq 0$. This gives

$$f(x) = 1 + x \lg x + \frac{x}{2} + xP(\lg x) - \frac{2}{1}x^2 + \frac{4}{3}x^3 - \cdots \; ,$$

where

$$P(t) = \frac{2\pi}{\ln 2} \sum_{n=1}^{\infty} \frac{\sin 2n\pi t}{\sinh(2n\pi^2/\ln 2)} \; .$$

The "Wobbles" Caused by $P(t)$

$P(t)$ is a very small periodic function:

$$|P(t)| < 7.8 \times 10^{-12}$$

for real t. In Lemma 3.1 of my 1976 paper, the term $xP(\lg x)$ was omitted.

Essentially, the poles off the real axis at

$$s = -1 \pm \frac{2\pi i n}{\ln 2} \; , \quad n = 1, 2, \ldots$$

were ignored[3].

Because the residues at the non-real poles are tiny (thanks to the sinh term in the denominator) numerical computations performed using single-precision floating-point arithmetic did not reveal the error.

[3] In fact, the incorrect result was obtained without using Mellin transforms. If I had used them I probably would have obtained the correct result!

Details of Corrections

The function $f(x)$ above is called $D_1(x)$ in my 1976 paper. In (3.29) of Lemma 3.1 of that paper, the expression for $D_1(x)$ is missing the term $xP(\lg x)$.

Equation (3.8) of the paper is (correctly)

$$F_n(x) = 1 + D_n(1/x) - D_n(x)$$

so in Corollary 3.2 the expression for $F_1(x)$ is missing a term $-xP(\lg x)$.

The statement following Corollary 3.2, that "In principle we could obtain $F_2(x)$, $F_3(x)$, etc in the same was as $F_1(x)$" is dubious because it is not clear how to handle the terms involving $P(\lg x)$.

Corollary 3.3, that $F_{n+1} \neq F_n$, is probably correct, but the proof given is incorrect because it assumes the incorrect form (5) for $F_n(x)$.

An Analogy

Ramanujan made a similar error when he gave a formula for $\pi(x)$ (the number of primes $\leq x$) which essentially ignored the residues of $x^s\zeta'(s)/\zeta(s)$ arising from zeros of $\zeta(s)$ off the real axis.

It is easier to work with

$$R(x) = \sum_{n=1}^{\infty} \frac{1}{n}\pi(x^{1/n})$$

than with $\pi(x)$. From $R(x)$ we can find $\pi(x)$ by Möbius inversion:

$$\pi(x) = \sum_{n=1}^{\infty} \frac{\mu(n)}{n} R(x^{1/n}) \ .$$

Riemann's formula

Riemann's explicit formula[4] for $R(x)$ is

$$R(x) = \text{li}\, x - \sum_{\rho} \text{li}\, x^{\rho} + \int_x^{\infty} \frac{dt}{(t^2-1)t\ln t} - \ln 2 \ .$$

The sum is over all the *complex* zeros ρ of the Riemann zeta function (summed in order of increasing $|\rho|$), and $\text{li}\, x$ is the logarithmic integral.

Ramanujan's error was essentially to ignore the sum over ρ.

[4] Stated by Riemann in 1859, and proved by Von Mangoldt in 1885.

Conclusion and Open Problems

Since Vallée's recent work, analysis of the average behaviour of the binary Euclidean algorithm has a rigorous foundation. However, some interesting open questions remain.

For example, does the binary Euclidean operator $\mathcal{B}_2$ have a unique positive dominant simple eigenvalue 1? Vallée has proved the corresponding result for her operator $\mathcal{V}_2$.

In order to estimate the speed of convergence of f_n to f (assuming f exists), we need more information on the spectrum of $\mathcal{B}_2$. What can be proved ? Preliminary numerical results indicate that the sub-dominant eigenvalue(s) are a complex conjugate pair:

$$\lambda_2 = \overline{\lambda}_3 = 0.1735 \pm 0.0884i \ ,$$

with $|\lambda_2| = |\lambda_3| = 0.1948$ to 4D.

Acknowledgements

Thanks to

- Don Knuth for encouraging me to correct and extend my 1976 results for the third edition of *Seminumerical Algorithms*,
- Brigitte Vallée for sharing her conjectures and results with me,
- Philippe Flajolet for his notes on Mellin transforms.

References

[1] B. C. Berndt, *Ramanujan's Notebooks*, Parts I-V, Springer-Verlag, New York, 1985, ...

[2] Adam W. Bojanczyk and Richard P. Brent, A systolic algorithm for extended GCD computation, *Comput. Math. Applic.* 14 (1987), 233–238. `http://www.comlab.ox.ac.uk/ oucl/work/richard.brent/pub/pub096.html`

[3] A. Borodin, J. von zur Gathen and J. Hopcroft, Parallel matrix and GCD computations, *Proc. 23rd Annual IEEE Symposium on Foundations of Computer Science*, IEEE, New York, 1982, 65–71.

[4] Richard P. Brent, Analysis of the Binary Euclidean Algorithm, *New Directions and Recent Results in Algorithms and Complexity*, (J. F. Traub, editor), Academic Press, New York, 1976, 321–355. `.../pub037.html`

[5] Richard P. Brent and H. T. Kung, Systolic VLSI arrays for linear-time GCD computation, in *VLSI 83* (F. Anceau and E. J. Aas, editors), North-Holland, Amsterdam, 1983, 145–154. `.../pub082.html`

[6] Richard P. Brent and H. T. Kung, A systolic VLSI array for integer GCD computation, in *ARITH-7, Proc. Seventh Symposium on Computer Arithmetic* (K. Hwang, editor), IEEE/CS Press, 1985. `.../pub077.html`

[7] Richard P. Brent, Twenty years' analysis of the binary Euclidean algorithm, *Millennial Perspectives in Computer Science: Proceedings of the 1999 Oxford – Microsoft Symposium in honour of Tony Hoare* (edited by J. Davies, A. W. Roscoe and J. Woodcock), Palgrave, New York, 2000, 41–53. `.../pub183.html`

[8] Hervé Daudé, Philippe Flajolet and Brigitte Vallée, An analysis of the Gaussian Algorithm for Lattice Reduction, *Proc. ANTS'94, Lecture Notes in Computer Science*, Vol. 877, Springer-Verlag, 1994, 144–158. Extended version in *Combinatorics, Probability and Computing* **6** (1997), 397–433.

[9] Philippe Flajolet and Brigitte Vallée, Continued Fraction Algorithms, Functional Operators and Structure Constants, *Theoretical Computer Science* **194** (1998), 1–34.

[10] G. H. Hardy, *Ramanujan: Twelve Lectures on Subjects Suggested by his Life and Work*, Cambridge University Press, Cambridge, 1940.

[11] Donald E. Knuth, *The Art of Computer Programming, Volume 2: Seminumerical Algorithms* (third edition). Addison-Wesley, Menlo Park, 1997.

[12] David Ruelle, *Thermodynamic formalism*, Addison Wesley, Menlo Park, 1978.

[13] Brigitte Vallée, Opérateurs de Ruelle–Mayer généralisés et analyse des algorithmes de Gauss et d'Euclide, *Acta Arithmetica* **81** (1997), 101–144.

[14] Brigitte Vallée, The complete analysis of the Binary Euclidean Algorithm, *Proc. ANTS'98, Lecture Notes in Computer Science*, Vol. 1423, Springer-Verlag, 1998, 77–94.

[15] Brigitte Vallée, Dynamics of the binary Euclidean algorithm: functional analysis and operators, *Algorithmica* **22** (1998), 660–685.