Summary
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. In this lecture I describe the classical and binary algorithms, and compare their worst case and average case behaviour. In particular, I correct some small but significant errors in the literature, discuss some recent results of Brigitte Vallée, and describe a numerical computation which verifies Vallée's conjecture to 44 decimal places.

Outline
- The classical Euclidean algorithm
  - The algorithm
  - Worst case
  - Continued fractions
  - Continuous model of Gauss
  - Results of Kuz’min, Lévy, et al
- The binary Euclidean algorithm
  - The algorithm
  - Worst case
  - Continuous model
  - Conjectured/empirical results
  - An error in the literature
  - Useful operators
  - Recent results of Vallée
  - Confirmation of a conjecture
  - Open problems

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

\( \text{Val}_2(u) \) denotes the dyadic valuation of the positive integer \( u \), i.e. the greatest integer \( j \) such that \( 2^j \mid u \).
The Classical Euclidean Algorithm

The Euclidean algorithm finds the greatest common divisor (GCD) of two positive integers \( m \) and \( n \). It is one of the best known of all algorithms. Knuth (§1.1) gives the algorithm as:

Algorithm E

E1. \( r \leftarrow m \mod n \)

E2. If \( r = 0 \) terminate with \( n \) as the result.

E3. Set \( m \leftarrow n, n \leftarrow r \), and go to E1

Of course, Euclid did not describe the algorithm in this way. In fact, it is not quite clear what Euclid intended at step E1. For a translation of Euclid’s description, see Knuth, §4.3.2.

The algorithm was probably known about 200 years before Euclid. Nevertheless, we shall call Algorithm E the “classical Euclidean algorithm” or just the “classical algorithm”.

Relation to Continued Fractions

Assume \( m \geq n \). The first execution of step E1 gives

\[
m = q \times n + r
\]

where \( q \) is the quotient and \( r \) is the remainder on division of \( m \) by \( n \). By definition,

\[
0 \leq r < n.
\]

Define \( n_0 = m, n_1 = n, n_2 = r = m \mod n \).

Suppose step E3 is executed \( k \) times. Then

\[
n_j = q_j \times n_{j+1} + n_{j+2}
\]

holds for \( j = 0, 1, \ldots, k \) and \( n_{k+2} = 0 \).

We can write this as

\[
\frac{n_j}{n_{j+1}} = q_j + \frac{1}{n_{j+2}}
\]

so, in the usual notation for continued fractions,

\[
\frac{m}{n} = q_0 + \frac{1}{q_1 + \frac{1}{q_2 + \cdots + \frac{1}{q_k}}}
\]

The Worst Case

We have seen that there is an intimate connection between the classical Euclidean algorithm for computing \( \text{GCD}(m, n) \) and the continued fraction expansion of the rational number \( m/n \).

The worst case for the classical algorithm occurs when all the quotients \( q_j \) are 1. This happens when the inputs are consecutive Fibonacci numbers. (These are defined by \( F_0 = 0, F_1 = 1, F_{j+2} = F_{j+1} + F_j \) for \( j \geq 0 \).)

For example,

\((m, n) = (F_6, F_5) = (8, 5) \rightarrow (5, 3) \rightarrow (3, 2) \rightarrow (2, 1)\)

Since \( F_k \sim \rho^k/\sqrt{5} \), where \( \rho = \frac{\sqrt{5} + 1}{2} \approx 1.618 \), the worst case number of iterations of the classical algorithm is

\[
\log_\rho N + O(1),
\]

where \( N = \max(m, n) \).
The Continuous Model of Gauss

To investigate the average behaviour of the classical algorithm, we can restrict attention to the case $0 < m < n$ (so $q_0 = 0$). We assume that $n = N$ is large and that $m$ is equally likely to take the values $\{1, 2, \ldots, N - 1\}$. Thus, $m/n$ will be approximately uniformly distributed in $(0, 1)$, and the sequence of quotients $q_1, q_2, \ldots$ will “look like” the quotients in the continued fraction expansion of a uniformly distributed random number.

Gauss’s Recurrence

Suppose $x_0 \in (0, 1)$, and $x_0$ has a continued fraction expansion

$$x_0 = 1/q_1 + 1/q_2 + \ldots + 1/(q_k + x_{k+1}),$$

where $q_1, \ldots, q_k$ are positive integers and $x_{k+1} \in (0, 1)$.

We can express the probability distribution function $F_{k+1}(x)$ of $x_{k+1}$ in terms of the distribution function $F_k(x)$ of $x_k$. Using the fact that $1/x_k = q_k + x_{k+1}$, we see that

$$F_{k+1}(x) = \Pr(0 \leq x_{k+1} \leq x)$$

$$= \sum_{q \geq 1} \Pr(q \leq 1/x_k \leq q + x)$$

$$= \sum_{q \geq 1} \Pr \left( \frac{1}{q + x} \leq x_k \leq \frac{1}{q} \right)$$

$$= \sum_{q \geq 1} \left( F_k \left( \frac{1}{q} \right) - F_k \left( \frac{1}{q + x} \right) \right)$$

The Limiting Distribution

To investigate average case behaviour in the continuous model, we assume $F_0(x) = x$ for $x \in (0, 1)$ (the uniform distribution of $m/n$), and consider $F_k(x)$ as $k \to \infty$.

If we assume that a limit distribution $F(x) = \lim_{k \to \infty} F_k(x)$ exists, then $F(x)$ satisfies the functional equation

$$F(x) = \sum_{q \geq 1} \left( F \left( \frac{1}{q} \right) - F \left( \frac{1}{q + x} \right) \right).$$

Gauss noticed the simple solution

$$F(x) = \log(1 + x).$$

However, Gauss was not able to prove convergence. It was eventually proved by Kuz’mín (1928). Sharper results were proved by Lévy (1929), Wirsing (1974), and Babenko (1978).

Babenko’s Theorem

The definitive result, due to Babenko (1978), is

$$F_k(x) = \log(1 + x) + \sum_{j \geq 2} \lambda_j \Psi_j(x),$$

where $|\lambda_2| > |\lambda_3| \geq |\lambda_4| \geq \cdots$, $-\lambda_2 = \lambda = 0.3036630028\ldots$

is called Wirsing’s constant, and the $\Psi_j(x)$ are certain analytic functions (see Knuth, §4.5.3 for more details).

The expected number of iterations is

$$\sim \frac{12 \ln 2}{\pi^2} \ln N \sim 0.8428 \ln N \sim 0.5842 \log N$$

which can be compared with

$$\sim \log_\phi N \sim 2.0781 \ln N \sim 1.4404 \log N$$

for the worst case.
Remainder of the Lecture

In the time remaining, I will describe what progress has been made towards a similar analysis of the binary Euclidean algorithm.

The Binary Euclidean Algorithm

The idea of the binary Euclidean algorithm is to avoid the “division” operation $r \leftarrow m \mod n$ of the classical algorithm, but retain $\mathcal{O}(\log N)$ 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

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

or equivalently

while even($u$) do $u \leftarrow u/2$

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

Definition of the Binary Algorithm

There are several almost equivalent ways to define the algorithm. For simplicity, let us 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. $t \leftarrow t/2^\text{Val}_2(t)$
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 Chi’u 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!
Another Formulation

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

**Algorithm V** { Assume \( u \leq v \) }

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

---

Continued Fractions

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

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

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:

\[
\begin{align*}
55 &= 9 + 2 \times 23 \\
 &= 9 + 2 \times (9 + 2 \times 7) \\
 &= 3 \times 9 + 4 \times 7
\end{align*}
\]

and on the next iteration,

\[
9 = 7 + 2 \times 1
\]

so

\[
\frac{55}{9} = 3 + \frac{4}{7},
\]

\[
\frac{9}{7} = 1 + \frac{2}{7},
\]

and finally

\[
\frac{9}{55} = \frac{1}{3 + \frac{4}{1 + \frac{2}{7}}},
\]

which 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

\[
\begin{align*}
v &\leftarrow u + 2^{k_1}v_1; \\
v_1 &\leftarrow u + 2^{k_2}v_2; \\
&\ldots \\
v_{m-1} &\leftarrow u + 2^{k_m}v_m
\end{align*}
\]

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 \( x_1 = v_\text{m}/u \) defined by

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

where \( a \) is an odd integer given by

\[
a = 1 + 2^{b_1} + 2^{b_2} + \ldots + 2^{b_k} - b_{m-1},
\]

and the exponent \( k \) is given by

\[
k = b_1 + \ldots b_m.
\]
Thus, the rational \( u/v \), for \( 1 \leq u < v \), has a unique binary continued fraction expansion of the form

\[
\frac{u}{v} = \frac{1}{a_1 + \frac{2^{k_1}}{a_2 + \frac{2^{k_2}}{\ddots + \frac{2^{k_r}}{a_r + \frac{2^{k_r}}{}}}}}
\]

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, \( k_1 + \cdots + k_r \).

\[ \text{Vallée’s Theorems} \]

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

\[ \text{The Worst Case} \]

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

\[ \lceil \log(u + v) \rceil \]

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

\[ \log(N) + O(1) \]

times.

\[ \text{Remark} \]

Even if step B2 is replaced by single-bit shifts

while even(\( t \)) do \( t \leftarrow t/2 \)

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

\[ \text{Extended Binary Algorithm} \]

It is possible to give an extended binary GCD algorithm which computes multipliers \( \alpha \) and \( \beta \) such that

\[ \alpha u + \beta v = \text{GCD}(u, v) \]

(Bojanczyk and Brent [2], 1987).

\[ \text{Systolic Binary Algorithm} \]

For hardware implementation, there is a systolic array variant of the binary GCD algorithm (Brent and Kung [5], 1985). This takes time \( O(\log N) \) using \( O(\log N) \) 1-bit processors and nearest-neighbour communication. The overall bit-complexity is \( O(\log N)^2 \).
A Heuristic Continuous Model

To analyze the expected behavior of Algorithm B, we can follow what Gauss did for the classical algorithm. This was first attempted in my 1976 paper [3] 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 $x \in [0, 1]$.

We assume that Val$2(t)$ takes the value $k$ with probability $2^{-k}$ at step B2. (Vallée does not make this assumption—we will discuss her rigorous analysis 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 Val$2(t) = k$ then $X = v/u$ is transformed to

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

It follows that $X' < x$ if

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

Thus, the recurrence for $G_n(x) = 1 - F_n(x)$ is

$$G_{n+1}(x) = \sum_{k \geq 1} 2^{-k} \left( G_n\left(\frac{1}{1 + 2^k x}\right) - G_n\left(\frac{1}{1 + 2^k x}\right) \right)$$

and $G_0(x) = 1 - x$ for $x \in [0, 1]$.

The Recurrence for $f_n$

Differentiating the recurrence for $G_n$ we obtain (formally) a recurrence for the probability density $f_n(x) = F'_n(x) = -G'_n(x)$:

$$f_{n+1}(x) = \sum_{k \geq 1} \left(\frac{1}{x + 2^k}\right)^2 f_n\left(\frac{x}{x + 2^k}\right) + \sum_{k \geq 1} \left(\frac{1}{1 + 2^k x}\right)^2 f_n\left(\frac{1}{1 + 2^k x}\right).$$

Operator Notation

The recurrence for $f_n$ may be written as

$$f_{n+1} = B_2 f_n,$$

where the operator $B_2$ is the case $s = 2$ of a more general operator $B_s$ which will be defined later.

Conjectured and Empirical Results

In my 1976 paper [3] 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 $f = B_2 f$ so $f$ is a “fixed point” of the operator $B_2$). Assuming the existence of $F$, it is shown in [3] that the expected number of iterations of Algorithm B is $\sim K \log N$ as $N \to \infty$, where $K = 0.705\ldots$ is a constant defined by

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

and

$$E_\infty = \int_0^1 \left( \sum_{n=0}^\infty \left(\frac{1 - x^n}{1 + (2^n - 1)x}\right) - \frac{1}{1 + x}\right) F(x) \, dx.$$
A Simplification
We can simplify the expression for $K$ to obtain

$$K = 2/b,$$

where

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

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.

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 [3], 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

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

If $f(x) = \sum_{k \geq 1} 2^{-k} g(2^k x)$ then the Mellin transform of $f$ is

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

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 $0 < \Re s < 1$, 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 $\Re s \leq 0$. This gives

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

where

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

An Error in the Literature
In (Brent, 1976) I claimed that, for all $n \geq 0$ and $x \in (0, 1]$,

$$F_n(x) = \alpha_n(x) \log(x) + \beta_n(x),$$

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

$$K = 0.70597124610194599986 \cdots$$

but I found

$$K = 0.70597124610191639152 \cdots$$

5–29

5–30

5–31

5–32
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 [3, Lemma 3.1], the term $x P(\log x)$ is omitted.

Essentially, we only considered poles on the real axis and ignored those at $s = -1 \pm 2\pi in/\ln 2$, $n = 1, 2, \ldots$

Because the residues at these 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.

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

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

than with $\pi(x)$. From $f(x)$ we can find $\pi(x)$ by M"{o}bius inversion:

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

Riemann’s formula

Riemann’s explicit formula\(^1\) for $f(x)$ is

$$f(x) = \log x - \sum_{\rho} \log |\rho| + \int_{2}^{x} \frac{dt}{t(t-1) \log t} - 2.$$  

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

Ramanujan’s error was essentially to ignore the sum over $\rho$.

Some Useful Operators

Operators $B_s, U_s, \check{U}_s, V_s$, useful in the analysis of the binary Euclidean algorithm, are defined on suitable function spaces by

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

$$\check{U}_s[f](x) = \left( \frac{1}{x} \right)^s U_s \left( \frac{1}{x} \right),$$

$$B_s = U_s + \check{U}_s,$$

$$V_s[f](x) = \sum_{k \geq 1} \sum_{a \in \mathbb{Z}} \left( \frac{1}{a + 2^k x} \right)^s f \left( \frac{1}{a + 2^k x} \right).$$

In these definitions $s$ is a complex variable, and the operators are called Ruelle operators [12]. They are linear operators acting on certain function spaces.

The case $s = 2$ is of particular interest. $B_2$ encodes the effect of one iteration of the inner “while” loop of Algorithm V, and $V_2$ encodes the effect of one iteration of the outer “while” loop.

\(^1\) Stated by Riemann in 1859, and proved by Von Mangoldt in 1885.
Relations between Operators

$B_2$ (denoted $T$) was introduced in my 1976 paper [5], and generalised to $B_s$ by Vallée. $V_s$ was introduced by Vallée. We shall call $B_2$ the binary Euclidean operator and $V_s$ Vallée’s operator. Not surprisingly, the operators are related, as the following Lemma and Theorem show.

**Lemma 1**

$$V_s = V_s U_s + U_s.$$

The following Theorem (which follows from the Lemma) gives a simple relationship between $B_s$, $V_s$ and $U_s$.

**Theorem 1**

$$(V_s - I)U_s = V_s (B_s - I).$$

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 $V_2[f](x)$. 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

$$V_2 U_2[f](x) + U_2[f](x),$$

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

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

$$K = \frac{4 \ln 2}{\pi^2 \lambda}. \quad (1)$$

Vallée proved the conjecture under the assumption that the operator $B_s$ satisfies a certain spectral condition. We have verified the conjectures numerically to 44 decimal places.

Recent Results of Vallée

Using her operator $V_s$, Vallée recently proved that

$$K = \frac{2 \ln 2}{\pi^2 g(1)} \sum_{a \neq 0} 2^{-|a|} G \left( \frac{1}{a} \right)$$

where $g$ is a nonzero fixed point of $V_2$ (i.e. $g = V_2 g \neq 0$) and $G(x) = \int_0^x g(t) \, dt$. This is yet another expression for $K$ (the only one which has been proved).

**Warning:** $G$ here is not the same as $G(x) = 1 - F(x)$! Unfortunately Knuth and Vallée use incompatible notation.

Because $V_s$ can be proved to have nice spectral properties, the existence and uniqueness (up to scaling) of $g$ can be proved rigorously.
Fixed Points of some Operators

It follows immediately from Theorem 1 that, if

\[ g = \mathcal{U}_2 f, \]

then

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

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 [15, Prop. 4] we know
that \( \mathcal{V}_2 \), acting on a certain Hardy space
\( \mathcal{H}^2(D) \), has a unique positive dominant simple
eigenvalue \( 1 \), so \( g \) must be (a constant multiple of)
the corresponding eigenfunction (provided
\( g \in \mathcal{H}^2(D) \)). Also, from the definitions of \( \mathcal{B}_2 \)
and \( \mathcal{V}_2 \), we have

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

which is useful for proving the consistency of
two of the expressions for \( K \) given above.

5–41

Numerical Results

Using an improvement of the “discretization
method” of [3], 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

\[ K = 0.705\,912\,646 \pm 0.016\,389\,52 \quad 0.344\,305\,852 \quad 8.817\,666\,677 \]
\[ \lambda = 0.397\,926\,841 \quad 8.831\,644\,067 \quad 0.707\,118\,02 \quad 6.598\,823\,598 \]
\[ K\lambda = 0.281\,304\,717 \quad 8.873\,535\,663 \quad 0.754\,309\,987 \quad 9.880\,385\,315 \]

These are believed to be correctly rounded
values.

One of Vallée’s conjectures 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).

5–42

Conclusion and Open Problems

Since Vallée’s recent work [14, 15], 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 [15, Prop. 4] 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:

\[ \lambda_2 = \bar{\lambda}_3 = 0.1735 \pm 0.0884i, \]

with \( |\lambda_2| = |\lambda_3| = 0.1948 \) to 4D.

5–43

Acknowledgements

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


Euclidean Algorithm, *New Directions and
Recent Results in Algorithms and Complexity*, (J. F. Traub, editor), Academic Press, New

VLSI arrays for linear-time GCD computation,
in *VLSI '83* (F. Aueceli and E. J. Aas, editors),

VLSI array for integer GCD computation, in
*ARITH-7, Proc. Seventh Symposium on
Computer Arithmetic* (K. Hwang, editor),

[6] Richard P. Brent, Further analysis of the
Binary Euclidean algorithm, in preparation.

5–45

[7] 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

[8] Philippe Flajolet and Brigitte Vallée,
Continued Fraction Algorithms, Functional
Operators and Structure Constants, *Theoretical

1812, Carl Friedrich Gauss Werke, Bd. X1,

Subjects Suggested by his Life and Work*,

Programming, Volume 2: Seminumerical
Algorithms* (third edition). Addison-Wesley,


5–46

généralisés et analyse des algorithmes de Gauss
101–144.

[14] Brigitte Vallée, The complete analysis of the
Binary Euclidean Algorithm, *Proc. ANTS'98,
Lecture Notes in Computer Science*, Vol. 1423,

Euclidean Algorithm: functional analysis and
operators, manuscript, Feb. 1998 (to appear in
*Algorithmica*).
http://www.info.unicaen.fr/~brigitte/
/Publications.bin-gcd.ps

5–47