

# ANALYSIS of EUCLIDEAN ALGORITHMS

An Arithmetical Instance of Dynamical Analysis

Dynamical Analysis :=

Analysis of Algorithms + Dynamical Systems

Brigitte VALLÉE (CNRS and Université de Caen, France)

Results obtained with :

Ali AKHAVI, Viviane BALADI, Jérémie BOURDON,  
Eda CESARATTO, Julien CLÉMENT, Benoît DAIREAUX,  
Philippe FLAJOLET, Loïck LHOTE, Véronique MAUME.

## Plan of the Talk

- I– The Euclid Algorithm, and the underlying dynamical system
- II– The other Euclidean Algorithms
- III– Probabilistic –and dynamical– analysis of algorithms
- IV– Euclidean algorithms : the underlying dynamical systems
- V– Dynamical analysis of Euclidean algorithms

## Plan of the Talk

I– The Euclid Algorithm, and the underlying dynamical system

II– The other Euclidean Algorithms

III– Probabilistic –and dynamical– analysis of algorithms

IV– Euclidean algorithms : the underlying dynamical systems

V– Dynamical analysis of Euclidean algorithms

The (standard) Euclid Algorithm: the grand father of all the algorithms.

On the input  $(u, v)$ , it computes the **gcd** of  $u$  and  $v$ ,  
together with the **Continued Fraction Expansion** of  $u/v$ .

## The (standard) Euclid Algorithm: the grand father of all the algorithms.

On the input  $(u, v)$ , it computes the **gcd** of  $u$  and  $v$ , together with the **Continued Fraction Expansion** of  $u/v$ .

$$u_0 := v; u_1 := u; u_0 \geq u_1$$

$$\left\{ \begin{array}{llll} u_0 & = & m_1 u_1 & + u_2 & 0 < u_2 < u_1 \\ u_1 & = & m_2 u_2 & + u_3 & 0 < u_3 < u_2 \\ \dots & = & \dots & + & \\ u_{p-2} & = & m_{p-1} u_{p-1} & + u_p & 0 < u_p < u_{p-1} \\ u_{p-1} & = & m_p u_p & + 0 & u_{p+1} = 0 \end{array} \right\}$$

$u_p$  is the **gcd** of  $u$  and  $v$ , the  $m_i$ 's are the **digits**.  $p$  is the **depth**.

## The (standard) Euclid Algorithm: the grand father of all the algorithms.

On the input  $(u, v)$ , it computes the **gcd** of  $u$  and  $v$ , together with the **Continued Fraction Expansion** of  $u/v$ .

$$u_0 := v; u_1 := u; u_0 \geq u_1$$

$$\left\{ \begin{array}{l} u_0 = m_1 u_1 + u_2 \quad 0 < u_2 < u_1 \\ u_1 = m_2 u_2 + u_3 \quad 0 < u_3 < u_2 \\ \dots = \dots + \\ u_{p-2} = m_{p-1} u_{p-1} + u_p \quad 0 < u_p < u_{p-1} \\ u_{p-1} = m_p u_p + 0 \quad u_{p+1} = 0 \end{array} \right\}$$

$u_p$  is the **gcd** of  $u$  and  $v$ , the  $m_i$ 's are the **digits**.  $p$  is the **depth**.

$$\text{CFE of } \frac{u}{v}: \quad \frac{u}{v} = \frac{1}{m_1 + \frac{1}{m_2 + \frac{1}{\ddots + \frac{1}{m_p}}}},$$

The underlying Euclidean dynamical system (I).

The trace of the execution of the Euclid Algorithm on  $(u_1, u_0)$  is:

$$(u_1, u_0) \rightarrow (u_2, u_1) \rightarrow (u_3, u_2) \rightarrow \dots \rightarrow (u_{p-1}, u_p) \rightarrow (u_{p+1}, u_p) = (0, u_p)$$

The underlying Euclidean dynamical system (I).

The trace of the execution of the Euclid Algorithm on  $(u_1, u_0)$  is:

$$(u_1, u_0) \rightarrow (u_2, u_1) \rightarrow (u_3, u_2) \rightarrow \dots \rightarrow (u_{p-1}, u_p) \rightarrow (u_{p+1}, u_p) = (0, u_p)$$

Replace the integer pair  $(u_i, u_{i-1})$  by the rational  $x_i := \frac{u_i}{u_{i-1}}$ .

The division  $u_{i-1} = m_i u_i + u_{i+1}$  is then written as

$$x_{i+1} = \frac{1}{x_i} - \left\lfloor \frac{1}{x_i} \right\rfloor \quad \text{or} \quad x_{i+1} = T(x_i), \quad \text{where}$$

$$T : [0, 1] \longrightarrow [0, 1], \quad T(x) := \frac{1}{x} - \left\lfloor \frac{1}{x} \right\rfloor \quad \text{for } x \neq 0, \quad T(0) = 0$$

## The underlying Euclidean dynamical system (I).

The trace of the execution of the Euclid Algorithm on  $(u_1, u_0)$  is:

$$(u_1, u_0) \rightarrow (u_2, u_1) \rightarrow (u_3, u_2) \rightarrow \dots \rightarrow (u_{p-1}, u_p) \rightarrow (u_{p+1}, u_p) = (0, u_p)$$

Replace the integer pair  $(u_i, u_{i-1})$  by the rational  $x_i := \frac{u_i}{u_{i-1}}$ .

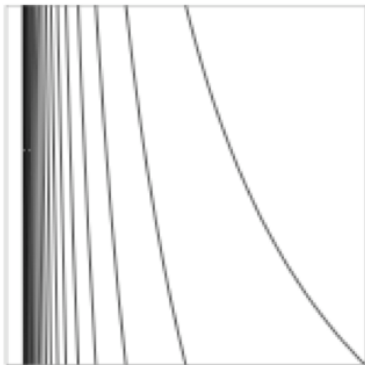
The division  $u_{i-1} = m_i u_i + u_{i+1}$  is then written as

$$x_{i+1} = \frac{1}{x_i} - \left\lfloor \frac{1}{x_i} \right\rfloor \quad \text{or} \quad x_{i+1} = T(x_i), \quad \text{where}$$

$$T : [0, 1] \longrightarrow [0, 1], \quad T(x) := \frac{1}{x} - \left\lfloor \frac{1}{x} \right\rfloor \quad \text{for } x \neq 0, \quad T(0) = 0$$

An **execution** of the Euclidean Algorithm  $(x, T(x), T^2(x), \dots, 0)$   
= A **rational trajectory** of the Dynamical System  $([0, 1], T)$   
= a **trajectory** that reaches **0**.

The dynamical system is a continuous extension of the algorithm.



$$T(x) := \frac{1}{x} - \left\lfloor \frac{1}{x} \right\rfloor$$

$$T_{[m]} := ]\frac{1}{m+1}, \frac{1}{m}[ \longrightarrow ]0, 1[,$$

$$T_{[m]}(x) := \frac{1}{x} - m$$

$$h_{[m]} := ]0, 1[ \longrightarrow ]\frac{1}{m+1}, \frac{1}{m}[$$

$$h_{[m]}(x) := \frac{1}{m+x}$$

## The Euclidean dynamical system (II).

A dynamical system with a denumerable system of branches  $(T_{[m]})_{m \geq 1}$ ,

$$T_{[m]} : ]\frac{1}{m+1}, \frac{1}{m}[ \longrightarrow ]0, 1[, \quad T_{[m]}(x) := \frac{1}{x} - m$$

The set  $\mathcal{H}$  of the inverse branches of  $T$  is

$$\mathcal{H} := \left\{ h_{[m]} : ]0, 1[ \longrightarrow ]\frac{1}{m+1}, \frac{1}{m}[; \quad h_{[m]}(x) := \frac{1}{m+x} \right\}$$

## The Euclidean dynamical system (II).

A dynamical system with a denumerable system of branches  $(T_{[m]})_{m \geq 1}$ ,

$$T_{[m]} : ]\frac{1}{m+1}, \frac{1}{m}[ \longrightarrow ]0, 1[, \quad T_{[m]}(x) := \frac{1}{x} - m$$

The set  $\mathcal{H}$  of the inverse branches of  $T$  is

$$\mathcal{H} := \left\{ h_{[m]} : ]0, 1[ \longrightarrow ]\frac{1}{m+1}, \frac{1}{m}[; \quad h_{[m]}(x) := \frac{1}{m+x} \right\}$$

The set  $\mathcal{H}$  builds **one step** of the CF's.

The set  $\mathcal{H}^n$  of the **inverse branches of  $T^n$**  builds CF's of **depth  $n$** .

The set  $\mathcal{H}^* := \bigcup \mathcal{H}^n$  builds **all the** (finite) CF's.

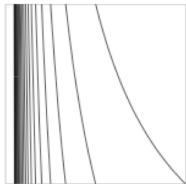
$$\frac{u}{v} = \frac{1}{m_1 + \frac{1}{m_2 + \frac{1}{\ddots + \frac{1}{m_p}}}} = h_{[m_1]} \circ h_{[m_2]} \circ \dots \circ h_{[m_p]}(0)$$

## The Euclidean dynamical system (III).

Density Transformer:

For a density  $f$  on  $[0, 1]$ ,  $\mathbf{H}[f]$  is the density on  $[0, 1]$  after one iteration of the shift

$$\mathbf{H}[f](x) = \sum_{h \in \mathcal{H}} |h'(x)| f \circ h(x) = \sum_{m \in \mathbb{N}} \frac{1}{(m+x)^2} f\left(\frac{1}{m+x}\right).$$

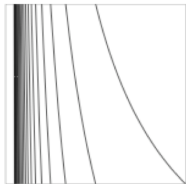


## The Euclidean dynamical system (III).

Density Transformer:

For a density  $f$  on  $[0, 1]$ ,  $\mathbf{H}[f]$  is the density on  $[0, 1]$  after one iteration of the shift

$$\mathbf{H}[f](x) = \sum_{h \in \mathcal{H}} |h'(x)| f \circ h(x) = \sum_{m \in \mathbb{N}} \frac{1}{(m+x)^2} f\left(\frac{1}{m+x}\right).$$



Transfer operator (Ruelle):

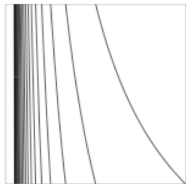
$$\mathbf{H}_s[f](x) = \sum_{h \in \mathcal{H}} |h'(x)|^s f \circ h(x).$$

## The Euclidean dynamical system (III).

Density Transformer:

For a density  $f$  on  $[0, 1]$ ,  $\mathbf{H}[f]$  is the density on  $[0, 1]$  after one iteration of the shift

$$\mathbf{H}[f](x) = \sum_{h \in \mathcal{H}} |h'(x)| f \circ h(x) = \sum_{m \in \mathbb{N}} \frac{1}{(m+x)^2} f\left(\frac{1}{m+x}\right).$$



Transfer operator (Ruelle):

$$\mathbf{H}_s[f](x) = \sum_{h \in \mathcal{H}} |h'(x)|^s f \circ h(x).$$

The  $k$ -th iterate satisfies:

$$\mathbf{H}_s^k[f](x) = \sum_{h \in \mathcal{H}^k} |h'(x)|^s f \circ h(x)$$

## Plan of the Talk

I– The Euclid Algorithm, and the underlying dynamical system

II– The other Euclidean Algorithms

III– Probabilistic –and dynamical– analysis of algorithms

IV– Euclidean algorithms : the underlying dynamical systems

V– Dynamical analysis of Euclidean algorithms

## Many variants of the Euclid Algorithm.

A Euclidean algorithm:=

any algorithm which performs a sequence of divisions  $v = mu + r$ .

There are various possible types of Euclidean divisions

– MSB divisions [directed by the Most Significant Bits]

shorten integers on the left,

and provide a remainder  $r$  smaller than  $u$ ,

(w.r.t the usual absolute value), i.e. with more zeroes on the left.

– LSB divisions [directed by the Least Significant Bits]

shorten integers on the right,

and provide a remainder  $r$  smaller than  $u$

(w.r.t the dyadic absolute value), i.e. with more zeroes on the right.

– Mixed divisions

shorten integers both on the right and on the left,

with new zeroes both on the right and on the left.

## Instances of MSB Algorithms.

– Variants according to the **position of remainder**  $r$ ,

**By Default:**  $v = mu + r$  with  $0 \leq r < u$

**By Excess:**  $v = mu - r$  with  $0 \leq r < u$

**Centered:**  $v = mu + \epsilon r$  with  $\epsilon = \pm 1, 0 \leq r \leq u/2$

## Instances of MSB Algorithms.

– Variants according to the **position of remainder**  $r$ ,

**By Default:**  $v = mu + r$  with  $0 \leq r < u$

**By Excess:**  $v = mu - r$  with  $0 \leq r < u$

**Centered:**  $v = mu + \epsilon r$  with  $\epsilon = \pm 1, 0 \leq r \leq u/2$

– **Subtractive** Algorithm :

A **division** with quotient  $m$  can be replaced by  $m$  **subtractions**

**While**  $v \geq u$  **do**  $v := v - u$

## An instance of a Mixed Algorithm.

The Subtractive Algorithm,

where the zeroes on the right are removed from the remainder defines the Binary Algorithm.

**Subtractive Gcd Algorithm.**

**Input.**  $u, v; v \geq u$

While ( $u \neq v$ ) do

    While  $v > u$  do

$v := v - u$

    Exchange  $u$  and  $v$ .

**Output.**  $u$  (or  $v$ ).

**Binary Gcd Algorithm.**

**Input.**  $u, v$  odd;  $v \geq u$

While ( $u \neq v$ ) do

    While  $v > u$  do

$k := \nu_2(v - u);$

$v := \frac{v - u}{2^k};$

    Exchange  $u$  and  $v$ .

**Output.**  $u$  (or  $v$ ).

The 2-adic valuation  $\nu_2$  counts the number of zeroes on the right

## An instance of a LSB Algorithm.

On a pair  $(u, v)$  with  $v$  odd and  $u$  even,

with  $\nu_2(u) = k$ , of the form  $u := 2^k u'$

the LSB division writes  $v = a \cdot u' + 2^k \cdot r'$ ,

with  $\nu_2(r') > \nu_2(u') = 0$  and  $\gcd(u, v) = \gcd(r', u')$ .

The pair  $(u', r')$  will be the new pair for the next step.

An execution of the LSB Algorithm:  
the Tortoise and the Hare

0	10001100101000001
1	111101011000000101000
2	11001001101101010000
3	110000110001010000000
4	10011000111100000000
5	11101001010100000000
6	11000001001000000000
7	10001000110000000000
8	10000010110000000000
9	1100000000000000
10	10000010000000000000
11	10001000000000000000
12	11000000000000000000
13	10000000000000000000

## Three main outputs for any Euclidean Algorithm

– the  $\gcd(u, v)$  itself

Essential in exact rational computations,

for keeping rational numbers under their irreducible forms

60% of the computation time in some symbolic computations

## Three main outputs for any Euclidean Algorithm

- the  $\gcd(u, v)$  itself
  - Essential in exact rational computations,
    - for keeping rational numbers under their irreducible forms
  - 60% of the computation time in some symbolic computations
- the Continued Fraction Expansion CFE  $(u/v)$ 
  - Often used directly in computation over rationals.

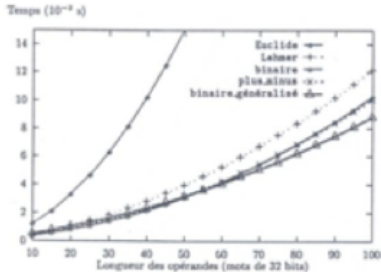
## Three main outputs for any Euclidean Algorithm

- the  $\gcd(u, v)$  itself
  - Essential in exact rational computations,
    - for keeping rational numbers under their irreducible forms
  - 60% of the computation time in some symbolic computations
- the Continued Fraction Expansion CFE  $(u/v)$ 
  - Often used directly in computation over rationals.
- the modular inverse  $u^{-1} \bmod v$ , when  $\gcd(u, v) = 1$ .
  - Extensively used in cryptography

A basic algorithm ... Perhaps the fifth main operation?

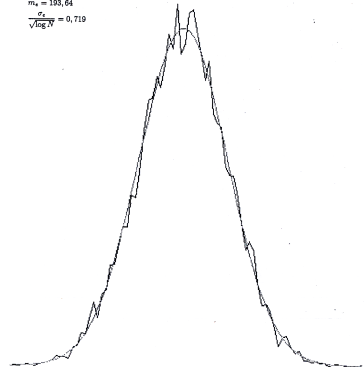
## Main algorithmic questions.

- Analyse the behaviour of these various Euclidean algorithms
- Compare them with respect to various costs  
and particularly the bit-complexity.



Experimental comparison  
of bit-complexities.

$$N = 10^{100}$$
$$S = 10^4$$
$$m_4 = 193,64$$
$$\frac{\sigma}{\sqrt{\log N}} = 0,719$$



A gaussian law  
for the number of steps?

## Explain the behaviour of algorithms

For instance, an execution of the LSB Algorithm : the **Tortoise** and the **Hare**

0	10001100101000001
1	111101011000000101000
2	11001001101101010000
3	110000110001010000000
4	10011000111100000000
5	11101001010100000000
6	11000001001000000000
7	10001000110000000000
8	10000010110000000000
9	1100000000000000
10	10000010000000000000
11	10001000000000000000
12	11000000000000000000
13	10000000000000000000

## Plan of the Talk

I– The Euclid Algorithm, and the underlying dynamical system

II– The other Euclidean Algorithms

III– Probabilistic –and dynamical– analysis of algorithms

IV– Euclidean algorithms : the underlying dynamical systems

V– Dynamical analysis of Euclidean algorithms

## Probabilistic Analysis of Algorithms

An **algorithm** with a set of **inputs**  $\Omega$ ,  
and a parameter (or a **cost**)  $C$  defined on  $\Omega$  which describes

- the **execution** of the algorithm (number of iterations, bit-complexity)
- or the **geometry** of the **output**  
(here: the continued fraction)

## Probabilistic Analysis of Algorithms

An **algorithm** with a set of **inputs**  $\Omega$ ,  
and a parameter (or a **cost**)  $C$  defined on  $\Omega$  which describes

- the **execution** of the algorithm (number of iterations, bit-complexity)
- or the **geometry** of the **output**  
(here: the continued fraction)

- Gather the inputs wrt to their **sizes** (here, their number of bits)

$$\Omega_n := \{(u, v) \in \Omega, \text{ size}(u, v) = n\}.$$

- Consider a **distribution on**  $\Omega_n$  (for instance the uniform distribution),
- Study the **cost**  $C$  on  $\Omega_n$  in a **probabilistic** way:
- Estimate the mean value of  $C$ , its variance, its distribution...  
in an **asymptotic** way (for  $n \rightarrow \infty$ )

## The main costs of interest for Euclidean Algorithms

- The additive costs, which depend on the digits

$$C(u, v) := \sum_{i=1}^p c(m_i)$$

if  $c = 1$ , then  $C :=$  the number of iterations

if  $c = \mathbf{1}_{m_0}$ , then  $C :=$  the number of digits equal to  $m_0$

if  $c = \ell$  (the binary length), then  $C :=$  the length of the CFE

## The main costs of interest for Euclidean Algorithms

- The additive costs, which depend on the digits

$$C(u, v) := \sum_{i=1}^p c(m_i)$$

if  $c = 1$ , then  $C :=$  the number of **iterations**

if  $c = \mathbf{1}_{m_0}$ , then  $C :=$  the number of digits equal to  $m_0$

if  $c = \ell$  (the binary length), then  $C :=$  the **length of the CFE**

- The bit complexity (not an additive cost)

$$C(u, v) := \sum_{i=1}^p \ell(u_i) \ell(m_i)$$

## The results (I)

### Previous results

- mostly in the **average**-case,
- **only** for the number of iterations, and **specific** to **particular** algorithms...
- well-described in Knuth's book (Tome II)

## The results (I)

### Previous results

- mostly in the **average**-case,
- **only** for the number of iterations, and **specific** to **particular** algorithms...
- well-described in Knuth's book (Tome II)

Heilbronn, Dixon, Rieger (70): **Standard** and **Centered** Alg.

Yao and Knuth (75): **Subtractive** Alg.

Brent (78): **Binary** Alg (**partly heuristic**),

Hensley (94) : A **distributional** study for the **Standard** Alg.

Stehlé and Zimmermann (05) : **LSB** Alg (**experiments**)

## The new results

With **Dynamical Analysis** method, our group [1995 → now ] obtains

## The new results

With **Dynamical Analysis** method, our group [1995 → now ] obtains

– a **complete classification** into two classes,

- the **Fast Class** = {Standard, Centered, Binary, LSB},
- the **Slow Class** = {By-Excess, Subtractive}.

## The new results

With **Dynamical Analysis** method, our group [1995 → now ] obtains

- a **complete classification** into two classes,
  - the **Fast Class** = {Standard, Centered, Binary, LSB},
  - the **Slow Class** = {By-Excess, Subtractive}.
- an **average-case** analysis of a broad **class of costs**,
  - all the **additive costs**
  - and also **the bit-complexity**.

## The new results

With **Dynamical Analysis** method, our group [1995 → now ] obtains

- a **complete classification** into two classes,
  - the **Fast Class** = {Standard, Centered, Binary, LSB},
  - the **Slow Class** = {By-Excess, Subtractive}.
- an **average-case** analysis of a broad **class of costs**,
  - all the **additive costs**
  - and also **the bit-complexity**.
- a **distributional** analysis of a subclass of the Fast Class,
  - the **Good Class** = {Standard, Centered}.

**Asymptotic gaussian laws** hold for:

- $P$ , and **additive costs** of **moderate** growth,
- the remainder size  $\log u_i$  for  $i \sim \delta P$ ,
- the **bit-complexity** of the extended Alg.

Main principles of Dynamical Analysis :=  
Analysis of Algorithms + Dynamical Systems

## 1– Interaction between the discrete world and the continuous world.

Three steps.

- (a) The **discrete** algorithm is extended into a **continuous** process.....
- (b) .... which is studied – more easily, using all the **analytic** tools.

## 1– Interaction between the discrete world and the continuous world.

Three steps.

- (a) The **discrete** algorithm is extended into a **continuous** process.....
- (b) .... which is studied – more easily, using all the **analytic** tools.
- (c) Returning to the **discrete algorithm**,  
with various principles of transfer from continuous to discrete.

The **discrete** data are of **zero measure** amongst the continuous data.

Main tools for probabilistic analysis of algorithms  
2– Generating functions ?

A classical tool : **Generating functions** of various types

$$A(z) := \sum_{n \geq 0} a_n z^n, \quad \hat{A}(z) := \sum_{n \geq 0} a_n \frac{z^n}{n!}, \quad \tilde{A}(s) := \sum_{n \geq 1} \frac{a_n}{n^s}$$

Directly used when the **distribution** of data does **not change** too much  
during the execution of the algorithm  
(for instance: the Euclid Algorithm on polynomials)

## Main tools for probabilistic analysis of algorithms

### 2- Generating functions ?

A classical tool : **Generating functions** of various types

$$A(z) := \sum_{n \geq 0} a_n z^n, \quad \hat{A}(z) := \sum_{n \geq 0} a_n \frac{z^n}{n!}, \quad \tilde{A}(s) := \sum_{n \geq 1} \frac{a_n}{n^s}$$

Directly used when the **distribution** of data does **not change** too much during the execution of the algorithm  
(for instance: the Euclid Algorithm on polynomials)

Here, this is not the case .... due to the existence of the **carries**

The study of the **dynamical system** underlying the algorithm explains how the **distribution** of data **evolves** during the execution of the algorithm.

It also describes the **behaviour** of the **generating functions** of costs...

## Main tools for probabilistic analysis of algorithms

### 3- Dynamical Analysis –main principles.

**Input.-** A discrete algorithm.

**Step 1.-** Extend the discrete algorithm into a continuous process, i.e. a dynamical system.  $(X, V)$   $X$  compact,  $V : X \rightarrow X$ , where the discrete alg. gives rise to particular trajectories.

## Main tools for probabilistic analysis of algorithms

### 3- Dynamical Analysis –main principles.

**Input.-** A discrete algorithm.

**Step 1.-** Extend the discrete algorithm into a continuous process, i.e. a dynamical system.  $(X, V)$   $X$  compact,  $V : X \rightarrow X$ , where the discrete alg. gives rise to particular trajectories.

**Step 2.-** Study this (continuous) dynamical system, via its generic trajectories. A main tool: the transfer operator.

## Main tools for probabilistic analysis of algorithms

### 3- Dynamical Analysis –main principles.

**Input.-** A discrete algorithm.

**Step 1.-** Extend the discrete algorithm into a continuous process, i.e. a dynamical system.  $(X, V)$   $X$  compact,  $V : X \rightarrow X$ , where the discrete alg. gives rise to particular trajectories.

**Step 2.-** Study this (continuous) dynamical system, via its generic trajectories. A main tool: the transfer operator.

**Step 3.-** Coming back to the algorithm: we need proving that the discrete trajectories behaves like the generic trajectories.

Use the transfer operator as a generating operator,

which generates itself ..... the generating functions

**Output.-** Probabilistic analysis of the Algorithm.

## A Euclidean Algorithm



## A Euclidean Algorithm



**Arithmetic** properties of the division



**Geometric** properties of the branches



**Spectral** properties of the transfer operator



**Analytical** properties of the generating function

## A Euclidean Algorithm



Arithmetic properties of the division



Geometric properties of the branches



Spectral properties of the transfer operator



Analytical properties of the generating function



Analysis of the Euclidean Algorithm

## Plan of the Talk

I– The Euclid Algorithm, and the underlying dynamical system

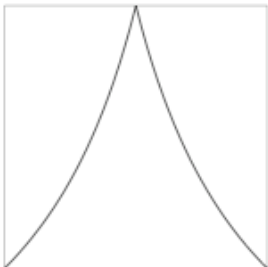
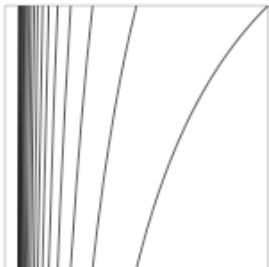
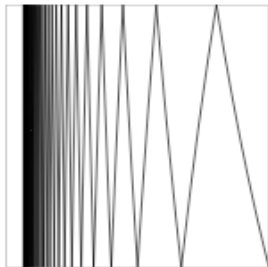
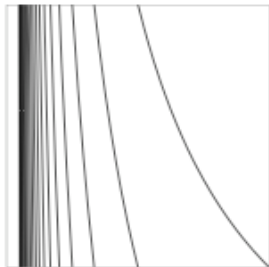
II– The other Euclidean Algorithms

III– Probabilistic –and dynamical– analysis of algorithms

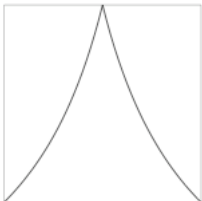
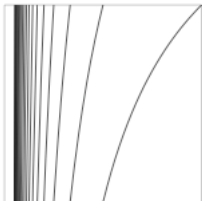
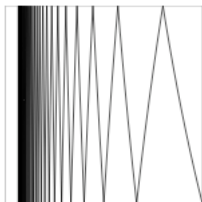
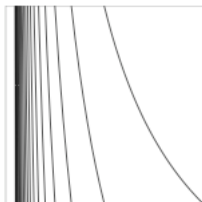
IV– Euclidean algorithms : the underlying dynamical systems

V– Dynamical analysis of Euclidean algorithms

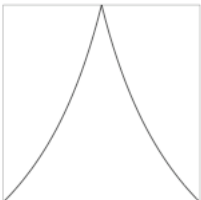
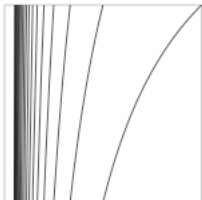
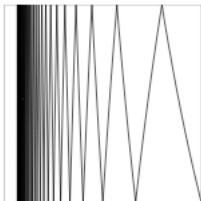
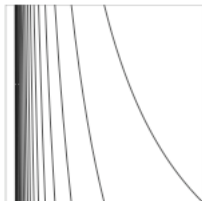
Four Euclidean dynamical systems (related to MSB divisions)



## Four Euclidean dynamical systems (related to MSB divisions)

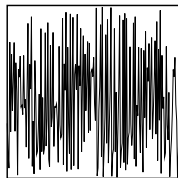


## Four Euclidean dynamical systems (related to MSB divisions)

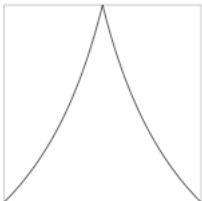
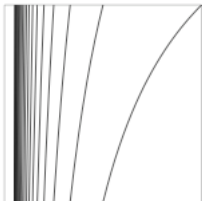
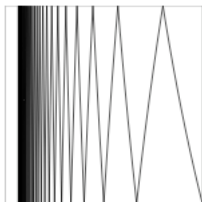
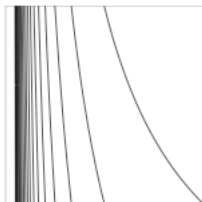


Two different classes

Fast Class

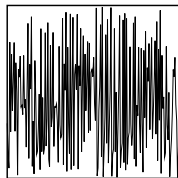


## Four Euclidean dynamical systems (related to MSB divisions)

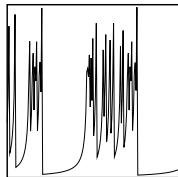


Two different classes

Fast Class



Slow Class



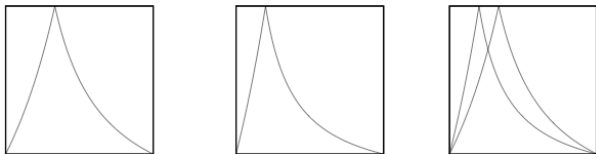
Two other Euclidean dynamical systems, related to mixed or LSB divisions

Two other Euclidean dynamical systems, related to mixed or LSB divisions

They are **probabilistic**: the **dyadic** valuation is viewed as a **random** variable

Two other Euclidean dynamical systems, related to mixed or LSB divisions

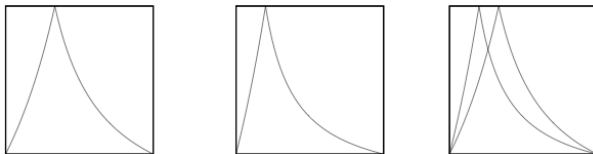
They are **probabilistic**: the **dyadic** valuation is viewed as a **random** variable



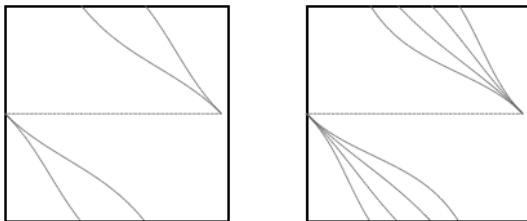
The DS relative to the Binary Algorithm

Two other Euclidean dynamical systems, related to mixed or LSB divisions

They are **probabilistic**: the **dyadic** valuation is viewed as a **random** variable



The DS relative to the Binary Algorithm



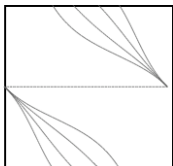
The DS relative to the LSB Algorithm

In all the cases (probabilistic or deterministic),  
the density transformer  $\mathbf{H}$  expresses the new density  $f_1$   
as a function of the old density  $f_0$ , as  $f_1 = \mathbf{H}[f_0]$ .  
It involves the set  $\mathcal{H}$

$$\mathbf{H}[f](x) := \sum_{h \in \mathcal{H}} \delta_h \cdot |h'(x)| \cdot f \circ h(x) \quad (\text{here, } \delta_h = \Pr[h])$$

With a cost  $c : \mathcal{H} \rightarrow \mathbf{R}^+$ , and a parameter  $s$ ,  
it gives rise to the weighted transfer operator

$$\mathbf{H}_s^{[c]}[f](x) := \sum_{h \in \mathcal{H}} \delta_h^s \cdot c(h) \cdot |h'(x)|^s \cdot f \circ h(x)$$



## Plan of the Talk

- I– The Euclid Algorithm, and the underlying dynamical system
- II– The other Euclidean Algorithms
- III– Probabilistic –and dynamical– analysis of algorithms
- IV– Euclidean algorithms : the underlying dynamical systems
- V– Dynamical analysis of Euclidean algorithms

## The Dirichlet series of cost $C$ .

If  $\Omega$  is the whole set of inputs, the Dirichlet generating function

$$S_C(s) = \sum_{(u,v) \in \Omega} \frac{C(u,v)}{|(u,v)|^{2s}} = \sum_{m \geq 1} \frac{c_m}{m^{2s}} \quad \text{with } c_m := \sum_{\substack{(u,v) \in \Omega \\ |(u,v)|=m}} C(u,v)$$

is used for expressing the mean value  $\mathbb{E}_n[C]$  of  $C$  on  $\Omega_n$ , since

$$\mathbb{E}_n[C] = \frac{1}{|\Omega_n|} \sum_{m|\ell(m)=n} c_m.$$

For the asymptotics of  $\mathbb{E}_n[C]$ ....

we need to obtain an alternative expression for  $S_C(s)$ , from which the position and the nature of singularities of  $S_C(s)$  become apparent.

## Relation between the transfer operator and the Dirichlet series.

Due to the fact that branches are LFT's,  
there is an alternative expression for the Dirichlet series

$$S_C(s) := \sum_{(u,v) \in \Omega} \frac{C(u,v)}{|(u,v)|^{2s}} = (I - \mathbf{H}_s)^{-1} \circ \mathbf{H}_s^{[c]} \circ (I - \mathbf{H}_s)^{-1}[1](\eta)$$

## Relation between the transfer operator and the Dirichlet series.

Due to the fact that branches are LFT's,  
there is an alternative expression for the Dirichlet series

$$S_C(s) := \sum_{(u,v) \in \Omega} \frac{C(u,v)}{|(u,v)|^{2s}} = (I - \mathbf{H}_s)^{-1} \circ \mathbf{H}_s^{[c]} \circ (I - \mathbf{H}_s)^{-1}[1](\eta)$$

as a function of two transfer operators : the **weighted** one

$$\mathbf{H}_s^{[c]}[f](x) = \sum_{h \in \mathcal{H}} \delta_h^s \cdot c(h) \cdot |h'(x)|^s \cdot f \circ h(x)$$

and the quasi-inverse  $(I - \mathbf{H}_s)^{-1}$  of the **plain** transfer operator  $\mathbf{H}_s$ ,

$$\mathbf{H}_s[f](x) := \sum_{h \in \mathcal{H}} \delta_h^s \cdot |h'(x)|^s \cdot f \circ h(x).$$

## Relation between the transfer operator and the Dirichlet series.

Due to the fact that branches are LFT's,  
there is an alternative expression for the Dirichlet series

$$S_C(s) := \sum_{(u,v) \in \Omega} \frac{C(u,v)}{|(u,v)|^{2s}} = (I - \mathbf{H}_s)^{-1} \circ \mathbf{H}_s^{[c]} \circ (I - \mathbf{H}_s)^{-1}[1](\eta)$$

as a function of two transfer operators : the **weighted** one

$$\mathbf{H}_s^{[c]}[f](x) = \sum_{h \in \mathcal{H}} \delta_h^s \cdot c(h) \cdot |h'(x)|^s \cdot f \circ h(x)$$

and the quasi-inverse  $(I - \mathbf{H}_s)^{-1}$  of the **plain** transfer operator  $\mathbf{H}_s$ ,

$$\mathbf{H}_s[f](x) := \sum_{h \in \mathcal{H}} \delta_h^s \cdot |h'(x)|^s \cdot f \circ h(x).$$

**Singularities** of  $s \mapsto (I - \mathbf{H}_s)^{-1}$  are related to **spectral properties** of  $\mathbf{H}_s$   
..... on a convenient functional space .... which depends on the DS (and the algo)...

## Expected spectral properties of $\mathbf{H}_s$

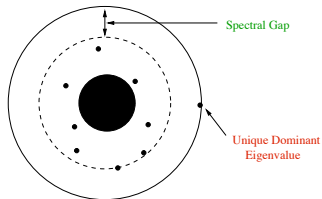
(i) *UDE* and *SG* for  $s$  near 1:

*UDE* – Unique dominant eigenvalue  $\lambda(s)$

with  $\lambda(1) = 1$

*SG* – Existence of a spectral gap

(ii) *Aperiodicity*: On the line  $\Re s = 1, s \neq 1$ ,  
the spectral radius of  $\mathbf{H}_s$  is  $< 1$



On *which* functional space?

The answer *depends* on the DS,  
and thus *on the algorithm*....

The functional spaces where the triple  $UDE + SG + Aperiodicity$  holds.

Algs	Geometry of branches	Convenient Functional space
Good Class (Standard, Centered)	Contracting	$C^1(\mathcal{I})$
Binary	Not contracting	The Hardy space $\mathcal{H}(\mathcal{D})$
LSB	Contracting on average	Various spaces: $C^0(J), C^1(J)$ Hölder $\mathbb{H}_\alpha(J)$
Slow Class (Subtractive, By-Excess)	An indifferent point	Induction + $C^1(\mathcal{I})$

In each case, the aperiodicity holds since the branches have not “all the same form”.

The triple  $UDE + SG + Aperiodicity$  entails good properties for  $(I - \mathbf{H}_s)^{-1}$ ,  
sufficient for applying Tauberian Theorems to  $S_C(s)$ .

$s = 1$  is the only pole  
on the line  $\Re s = 1$



Expansion near the pole  $s = 1$   
$$(I - \mathbf{H}_s)^{-1} \sim \frac{a}{s - 1}$$

$s=1$

Half-plane of convergence  $\Re s > 1$

No hypothesis needed  
on the half-plane  $\Re s < 1$ .

## A Euclidean Algorithm



Arithmetic properties of the division



Geometric properties of the branches



Spectral properties of the transfer operator



Analytical properties of the generating function



Analysis of the Euclidean Algorithm