Effective fractal dimension theory: exploring the extreme cases (II)

Elvira Mayordomo

Universidad de Zaragoza, Iowa State University

August 24th 2017

Left out Monday

- Open problems:
  - Partially complete problems
  - BPP sources
- References

Today

0. Introduction of effective dimension
1. Resource-bounded Hausdorff dimension for Complexity Classes
2. **Compression and dimension for low resource bounds. Very effective construction of a normal sequence**
3. Looking back at fractal geometry, other metric spaces

Compression characterizations of effective measure

- **Constructive Hausdorff dimension can be entirely defined using Kolmogorov complexity**

Theorem

For every $A \subseteq \{0,1\}^\infty$, $\mathrm{cdim}(A) = \inf_{x \in A} \frac{\mathrm{K}(x{\upharpoonright}n)}{n}$.

For a finite string w, $\mathrm{K}(w)$ is the length of the shortest description from which w can be computably recoverered

Space-bounded Kolmogorov complexity characterizations

$$\mathrm{KS}^f(w) = \min\{|p| \mid U(p) = w \text{ in space } f(|w|)\}$$

Theorem

For every $A \subseteq \{0,1\}^\infty$,
$\dim_{\mathrm{pspace}}(A) = \inf_{q \textit{ polynomial}} \inf_{x \in A} \frac{\mathrm{KS}^q(x\restriction n)}{n}$.

What about time?

- Time-bounded Kolmogorov complexity is hard to work with due to invertibility issues
- p-dimension (predictability) can be characterized in terms of a class of polynomial time reversible compressors: *compressors that do not start from scratch*
- I will leave out the technical definition, but for instance a compressor C for which $C(w)$ and $C(wu)$ have a common prefix of length at least $|C(w)| - O(\log(|w|))$ does not start from scratch

Theorem

p-dimension is exactly the best compression rate achievable through polynomial-time compressors that do not start from scratch

Pushdown dimension

We consider BPD the set of pushdown machines that work with a bounded number of λ-transitions per input symbol

Theorem

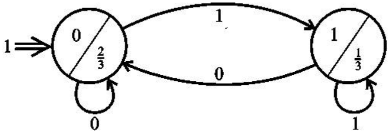
BPD-dimension is exactly the best compression rate achievable through BPD-compressors

Still open for general PD-computation

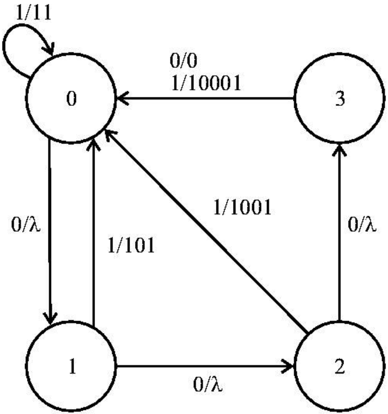
Finite state dimension

We considered the case of Finite-State computation

$\dim_{\mathrm{FS}}(A) = \inf\{s \mid$ there is a Finite State s-gale that succeeds on $A\}$



Finite state compression



The input can be recovered given the output and the final state

Finite state dimension characterization

Theorem

Finite-state dimension is exactly the best compression rate achievable through finite-state compressors, that is,

$$\dim_{\mathrm{FS}}(x) = \inf_{C\ \mathrm{FS}-comp} \liminf_{n} \frac{|C(x \upharpoonright n|}{n}$$

FS-compressors and Lempel-Ziv algorithm

- **Lempel-Ziv algorithm** subsumes FS-compressors:

$$\rho_{LZ}(x) = \liminf \liminf_{n} \frac{|LZ(x \upharpoonright n|}{n}$$

Theorem

For every $x \in \{0,1\}^\infty$

$$\rho_{LZ}(x) \leq \dim_{\mathrm{FS}}(x)$$

- Lempel-Ziv algorthm is universal for FS dimension/compression
- It is known that there are sequences for which $\rho_{LZ}(x) < \dim_{\mathrm{FS}}(x)$

Comparison among different levels

Theorem

PD-compression is incomparable with the Lempel-Ziv compression algorithm:

- *There are sequences for which PD-compression is better than LZ.*
- *There are sequences for which Lempel-Ziv compression is better than PD.*

- There is a FS-random sequence that is not PD-random (note: FS-random is equivalent to FS-dimension 1)
- There is a sequence such that $\dim_{\mathrm{PD}}(x) < \dim_{\mathrm{FS}}(x) < 1$

Open question

- Is PD-dimension 1 different from FS-dimension 1?
- Can PD-dimension be characterized in terms of compression?

Open questions

- There are characterizations of effective fractal dimension in terms of Kolmogorov complexity/compressibility at the most and least restricted computation levels
- They happen for completely different reasons
- Understanding what happens at intermediate levels can have useful applications for learning/compression
- Understanding what happens with FS-dimension/randomness may be useful for number theory

References for compression and dimension

- E. Mayordomo. A Kolmogorov complexity characterization of constructive Hausdorff dimension. Information Processing Letters, 84(1):1-3, 2002.
- M. López-Valdés and E. Mayordomo. Dimension is compression. In Proceedings of the 30th International Symposium on Mathematical Foundations of Computer Science, pages 676-685. Springer-Verlag, 2005.
- P. Albert, E. Mayordomo, and P. Moser. Bounded Pushdown dimension vs Lempel Ziv information density. Computability and Complexity pp. 95-114, Essays Dedicated to Rodney G. Downey on the Occasion of His 60th Birthday (Day, A., Fellows, M., Greenberg, N., Khoussainov, B., Melnikov, A., Rosamond, F. (Eds.)) Lecture Notes in Computer Science book series (LNCS, volume 10010), 2017.

References for compression and dimension

- J. J. Dai, J. I. Lathrop, J. H. Lutz, and E. Mayordomo. Finite-state dimension. Theoretical Computer Science, 310(1-3):1-33, 2004.
- D. Doty and J. Nichols. Pushdown dimension. Theoretical Computer Science, 381(1-3):105-123, 2007.
- W. Merkle, J. Reimann. On selection functions that do not preserve normality. Theory of Computing Systems, 39(5):685697, 2006.

Very effective construction of an absolutely normal sequence

- At the lowest resource-bounded level, FS, dimension meets number theory
- Sequences with FS-dimension 1 are exactly Borel normal sequences
- FS-dimension is not closed under base change
- Can we use a constructive probabilistic method to construct an absolutely normal sequence?

Normal numbers

Borel, 1909:

- A real number α is **normal** in base b $(b \geq 2)$ if, for every finite sequence w of base-b digits,

$$\lim_n \frac{\mathrm{N}_\alpha(w, n)}{n} = \frac{1}{b^{|w|}}$$

  the asymptotic, empirical frequency of w in the base-b expansion of α is $b^{-|w|}$.

- α is **absolutely normal** if it is normal in every base $b \geq 2$.

Theorem (Cassels 1959, Schmidt 1960)

There exist a number that is normal in base 2 but not in base 3.

Examples of normal numbers

- Champernowne's sequence

$$0.123456\ldots$$

- Copeland-Erdös sequence

$$0.235711\ldots$$

Pretty far from natural

Absolutely Normal numbers

Theorem (Borel)

Almost every real number (i.e., every real number outside a set of Lebesgue measure 0) is absolutely normal.

- Computer analyses of the expansions of π, e, $\sqrt{2}$, $\ln 2$, and other irrational numbers that arise in common mathematical practice **suggest** that these numbers are absolutely normal.
- No such "natural" example of a real number has been proven to be normal in *any* base, let alone absolutely normal.
- The **conjectures that every algebraic irrational is absolutely normal** and that π is absolutely normal are especially well known open problems.
- In cryptographic applications constants such as π and $\sqrt{2}$ are used and expected to be "somehow random" (nothing up my sleeve numbers)

Computing absolutely normal numbers

- We are interested in the **complexity of** explicitly **computing** an absolutely normal real number
- Sierpinski and Lebesgue gave explicit constructions of absolutely normal numbers in 1917 (intricate limiting processes, no complexity or insight into the nature of the numbers constructed)
- Turing (1936, unpublished) gave a constructive proof that almost all real numbers are absolutely normal and then *derived* constructions of absolutely normal numbers from this proof.

Turing vision

- We believed that Schmidt (1960) was the first to construct absolutely normal numbers
- But the most surprising part was Turing's idea of effective measure and its application as an effective probabilistic method
- As analysed by Figueira, Becher and Picci (2007) Turing's unpublished note shows is that the set of non-normal numbers has computable measure 0
- The formalization of effective measure and randomness did not come until the sixties: Martin-Löf (paper 1966), Von-Mises, Solomonoff (1960), Kolmogorov, ...
- We now know that normality is a type of randomness

Computing absolutely normal numbers

- (Becher, Heiber, and Slaman 2013, simultaneous work from other authors) Algorithm that computes an absolutely normal number in polynomial time.
- Specifically, they compute the binary expansion of an absolutely normal number x, with the nth bit of x appearing after $O(n^2 \mathrm{polylog}(n))$ steps.
- Here we present a new algorithm that computes an absolutely normal in **nearly linear time**. Our algorithm computes the binary expansion of an absolutely normal number x, with the nth bit of x appearing after $O(n \mathrm{polylog}(n))$ steps.

Note: The term "nearly linear time" was introduced by Gurevich and Shelah (1989). While linear time computability is very model-dependent, nearly linear time is very robust.

Gales and martingales in base b

- $\Sigma_b = \{0, \ldots, b-1\}$ the base b alphabet
- Σ_b^* are finite sequences, Σ_b^∞ infinite sequences
- For $s \in [0, \infty)$, an s-gale is a function $\Sigma_b^* \to [0..\infty)$ such that for $w \in \Sigma_b^*$

$$d(w) = \frac{d(w0) + d(w1)}{b^s}$$

- A **martingale** is a function $d : \Sigma_b^* \to [0..\infty)$ with the fairness property, for every finite sequence w,

$$d(w) = \frac{\sum_{i \in \Sigma_b} d(wi)}{b}$$

- The success set of an s-gale d is

$$S^\infty[d] = \left\{ x \in \Sigma_b^\infty \,\middle|\, \limsup_n d(x \upharpoonright n) = \infty \right\}$$

- **Notice that if d is an s-gale then $d'(w) = b^{(1-s)|w|} d(w)$ is a martingale**

Finite-state randomness

Definition

x is **FS random** is no finite automata computable martingale succeeds on x

Notice that if $\dim_{\mathrm{FS}}(x) < 1$ then x is not FS-random

Normality and Finite-state randomness

- If x is the base b representation of a non-normal number, w is a finite string that is "unbalanced" in x, for instace i.o. w appears more often than it should, a finite automata can bet a bit more than its fair share and make infinite money ...
- Clearly FS random sequences are representations of base b normal numbers
- Even better **FS-random** $=$ **normal** –Schnorr and Stimm (1972)

Finite-State dimension

Schnorr and Stimm (1972) implicitly defined **finite-state martingales** and proved that every sequence $S \in \Sigma_b^\infty$ obeys this dichotomy:

1. If S is b-normal, then no finite-state base-b martingale succeeds on S. (In fact, every finite-state base-b martingale decays exponentially on S.)
2. If S is not b-normal, then some finite-state base-b martingale **succeeds exponentially** on S.

Using dimension terminology

1. If S is b-normal, then S is FS-random.
2. If S is not b-normal, then $\dim_{\mathrm{FS}}(S) < 1$.

Therefore **FS-dimension 1** $=$ **normal**

Remember ...

- **Objective** Compute a (provably) absolutely normal number $x \in (0,1)$ **fast**.
- Absolutely normal number means that is normal in every base
- We need to construct a single real number that is b-normal for every base b
- We will use Lempel-Ziv algorithm that is universal for FS-compressors **in a single base**

Lempel-Ziv martingales

Feder (1991) implicitly defined the **base-b Lempel-Ziv martingale** $d_{\mathrm{LZ}(b)}$ and proved that it is at least as successful **on every sequence** as every finite-state martingale.

$\therefore$ if $S \in \Sigma_b^\infty$ is not normal, then $\dim_{d_{\mathrm{LZ}(b)}}(S) < 1$.

$\therefore$ $x \in (0,1)$ is absolutely normal if none of the martingales $d_{\mathrm{LZ}(b)}$ succeed **exponentially** on the base-b expansion of x.

Moreover, $d_{\mathrm{LZ}(b)}$ has a fast and beautiful theory.

Celebrated Lempel-Ziv compression algorithm and martingale can be both computed very efficiently (time very close to linear)

Lempel-Ziv martingales

How $d_{\mathrm{LZ}(b)}$ works:
Parse $w \in \Sigma_b^*$ into distinct **phrases**, using a growing tree whose leaves are all of the previous phrases.
At each step, bet on the next digit in proportion to the number of leaves below each of the b options.

Base change notation

- For a real x, $\mathrm{seq}_b(x) \in \Sigma_b^\infty$ is the base-b representation of x
- For a sequence $S \in \Sigma_b^\infty$, $\mathrm{real}_b(S) \in [0,1]$ is the real number represented by S

How to construct an absolutely normal number

- For each base b, we need to construct x such that b-Lempel-Ziv martingale does not succeed on x
- We need to construct a single real number that is b-normal for every base b
- It suffices to translate b-Lempel-Ziv martingale into base 2 (very efficiently)
- We need a martingale $d : \Sigma_2^* \to [0, \infty)$ that succeeds on base 2 representations of the numbers for which b-Lempel-Ziv martingale succeeds
- For this translation to be possible (**and efficient**) the martingale must be quite well behaving ...

How to construct an absolutely normal number

1. Transform b-Lempel-Ziv martingale into a better behaving and still efficient martingale that still succeeds on not b-normal sequences
2. Efficiently change base for the resulting martingale
3. Efficiently combine all resulting martingales into one
4. Diagonalize resulting martingale

Savings Accounts, strong success

- The value of Lempel-Ziv martingale $d_{\mathrm{LZ}(b)}$ on a certain infinite string S can fluctuate a lot
- This makes base change more complicated (and time consuming)
- We use the notion of "savings account" here, we are looking at an alternative martingale that **keeps money aside for the bad times to come**

The strong success set of an s-supergale d is

$$S^{\infty}_{\mathrm{str}}[d] = \left\{ x \in \{0,1\}^{\infty} \,\middle|\, \lim_{n} d(x \upharpoonright n) = \infty \right\}$$

Savings Accounts, strong success

- We construct a new martingale d'_b that is a conservative version of $d_{\mathrm{LZ}(b)}$
- d'_b strongly succeeds at least on non-b-normal sequences

$$\{S \,|\dim_{LZ}(S) < 1\} \subseteq S^{\infty}_{\mathrm{str}}[d'_b]$$

- d'_b can be computed in nearly linear time
- If $S \notin S^{\infty}_{\mathrm{str}}[d'_b]$ then S is b-normal

Base Change

- We want an absolutely normal real number x, that is, the base b representation $seq_b(x)$ is not in $S^\infty[d'_b]$
- For this we convert d'_b into a base-2 martingale $d_b^{(2)}$ succeeding on the **base-2 representations of the reals with base-b representation in $S^\infty_{\mathrm{str}}[d'_b]$**
- Again, $d_b^{(2)}$ succeeds on $\mathrm{seq}_2(\mathrm{real}_b(S^\infty_{\mathrm{str}}[d'_b])$

$$\mathrm{real}_b(S^\infty_{\mathrm{str}}[d'_b]) \subseteq \mathrm{real}_2(S^\infty_{\mathrm{str}}[d_b^{(2)}])$$

- We use Carathéodory construction to define measures
- Computing in nearly linear time is also delicate
- In fact our computation $\widehat{d_b^{(2)}}$ approximates slowly $d_b^{(2)}$

$$|\widehat{d_b^{(2)}}(y) - d_b^{(2)}(y)| \leq \frac{1}{|y|^3}$$

Absolutely Normal Numbers

- From previous steps we have a family of martingales $(d_b^{(2)})_b$ so that $d_b^{(2)}$ **succeeds on base-2 representations of non-b-normal sequences**
- For each b we have a nearly linear time computation $\widehat{d_b^{(2)}}$
- We want to construct $S \notin S^\infty[d_b^{(2)}]$ for every b
- Nearly linear time makes it painful to construct a martingale d for the union of $S^\infty[d_b^{(2)}]$
- Then we diagonalize over d to construct S

Martingale diagonalization

- For a martingale d, how to construct x such that d martingale does not succeed on x (with time similar to the computation time for d)?
- Recursive construction, if we have the prefix $x \upharpoonright n$ choose the next symbol i such that

$$d(x \upharpoonright ni)$$

  is the minimum over all possible symbols
- By the fairness condition of a martingale

$$d(w) = \frac{\sum_{i \in \Sigma_b} d(wi)}{b}$$

  d does not succeed on the resulting x
- Time is $n \cdot t(n)$ if d is computable in time $t(n)$

Time bounds ...

- All the steps were performed in nearly linear time on a common **time bound independent of base** b
- Many technical details were simplified in this presentation ... please read paper

Base invariance

- Normality corresponds exactly to the lowest level of algorithmic randomness, Finite-State randomness
- Finite-State randomness and Finite-State dimension are not closed under base change
- p-dimension and p-randomness are closed under base change
- What about intermediate levels, PD, LZ, nearly linear time?

Conclusions

- Lots of remaining questions,
  - can we substitute “suspected” absolute normal numbers by proven absolutely normal numbers in Cryptography?
  - “biased-normality”? (based on FS-dimension)
  - Tight complexity for the operation of base change
  - The algorithm of Becher, Heiber, and Slaman’s has nearly quadratic time but (apparently) a much lower discrepancy. Can we improve our discrepancy while maintaining nearly linear time?

References for construction of absolutely normal numbers

- J. H. Lutz and E. Mayordomo, Computing absolutely normal numbers in nearly linear time, submitted. (arxiv 1611.05911)
- V. Becher, P.A. Heiber, and T. Slaman, A polynomial-time algorithm for computing absolutely normal numbers, Information and Computation 232: 1–9, 2013.
- C. Aistleitner, V. Becher, A.-M. Scheerer, and T. Slaman, On the construction of absolutely normal numbers, 2017, to appear in Acta Arithmetica. (arXiv 1707.02628)
- V. Becher, S.Figueira, and R. Picchi, Turing's unpublished algorithm for normal numbers, Theoretical Computer Science 377: 126-138, 2007.

Next lecture

0. Introduction of effective dimension
1. Resource-bounded Hausdorff dimension for Complexity Classes
2. Compression and dimension for low resource bounds. Very effective construction of a normal sequence
3. **Looking back at fractal geometry, other metric spaces**