Clustering and Causality Inference Using Algorithmic Complexity

Marion Revolle, François Cayre, and Nicolas Le Bihan
GIPSA-Lab / CNRS
Email: first.last@gipsa-lab.grenoble-inp.fr

***Abstract*—We present a set of algorithmic complexity estimates. We derive a normalized semi-distance that is shown to outperform the state-of-the-art. We also propose estimators for causality inference on directed acyclic graphs. Illustrative applications include clustering of human writing systems and causality assessment on novel drafts.**

I. Introduction

Compared to classical probabilistic information theory, algorithmic information theory (AIT) does not require estimating probability density/mass functions. It is solely based on Kolmogorov complexity, which treats the data as is, and not as being the realization of an underlying model. However, it turns out that Kolmogorov complexity is not computable on a universal Turing machine. One has to resort to approximations thereof. Kolmogorov complexity is the size of the shortest program able to reproduce the input when ran on a Turing machine. While deeply rooted in source coding, we shall propose estimates based on the Lempel-Ziv family of algorithms only, discarding the entropy coding stage usually found in a modern compressor.

Our goal is to propose estimates for information-theoretic quantities, namely self-, conditional and joint complexity estimates of symbol sequences. Given these estimates, we show how to derive a normalized semi-distance and estimates for causality inference on directed acyclic graphs. We shall compare the performance of our semi-distance with the Normalized Compression Distance (NCD) [1].

II. Related works

The pioneering works in the area [1], [2] have targeted the design of an algorithmic distance between two sequences of symbols. Let x and y be two sequences defined respectively over the alphabets $\mathcal{A}_x$ and $\mathcal{A}_y$. Let $K(x|y)$ be the conditional Kolmogorov complexity of the sequence x when the sequence y is known. It can be shown [2] that, however not computable, the following expression is a distance between x and y:

$$E_1(x,y) = \max\{K(x|y), K(y|x)\}. \tag{1}$$

In order to compare objects of different sizes, the following normalization, known as the Normalized Information Distance (NID), is the most appropriate choice [1]:

$$\mathrm{NID}(x,y) = \frac{\max\{K(x|y), K(y|x)\}}{\max\{K(x), K(y)\}}, \tag{2}$$

where $K(x)$ denotes the Kolmogorov complexity of x.

Given the definition of Kolmogorov complexity, it seems reasonable to approximate it using a compressor C. Another approximation relates to the conditional complexity [1]:

$$K(x|y) \approx K(xy) - K(y), \tag{3}$$

where xy denotes the concatenation of sequences x and y.

From there, one can easily derive a "distance" measure, known as the Normalized Compression Distance (NCD) [1]:

$$\mathrm{NCD}(x,y) = \frac{C(xy) - \min\{C(x), C(y)\}}{\max\{C(x), C(y)\}}. \tag{4}$$

However extremely simple to compute once a compressor is available, this approach suffers from several shortcomings:

- the compressor itself has builtin limitations (*e.g.* the window size in LZ77-based compressors: it is 32KiB for `DEFLATE` [3], hence one could theoretically compare sequences of maximum 16KiB due to concatenation);
- the approximation of Eq. 3 does not ensure that *only* sequences from y will be used to describe x, this is also true for coders based on LZMA with a larger window (4GiB for LZMA [4]).

The practical consequences of using a particular compressor are summarized in [5].

The rest of this paper is devoted to proposing estimates that do not suffer from these shortcomings.

III. What conditional information?

It is certainly worth recalling that a LZ77 coder works by finding references to subsequences it has already seen, provided these references fall within a finite sliding window of past symbols. Hence, the position of the current input pointer plays a crucial role: everything beyond the size of the window in the past is forgotten and cannot be used to find subsequences. If a subsequence cannot be found in the window, then a literal is emitted.

Our proposal is twofold:

1) to use a semi-infinite sliding window: at any step of the LZ77 encoding, a subsequence can be referenced arbitrarily far from the past.
2) to generalize LZ77 coding, Ziv-Merhav universal classification [6] and two other previously undefined settings into a framework that allows to easily express our estimates.

The key issue in designing such estimates is how conditional information is taken into account (second point above). For

ISBN 978-0-9928626-7-1

example, the conditional information in our LZ77 coding is the entire past of the sequence x being encoded. For the Ziv-Merhav relative coder [6], it is the entire past of the known sequence y up to the current position when encoding the sequence x.

Hence, we need to parameterize $\mathcal{R}$: the (possibly infinite) sequence(s) in which references to subsequences can be made. We have the following four cases, updated for any position of the input sequence being encoded:

1) $y|x$: $\mathcal{R}$ is only the past of x:
   This models the usual LZ77 operating mode when $x = y$ (needed in Sec. IV-C), and $\mathcal{A}_\mathcal{R} = \mathcal{A}_x$;
2) $y|^+x$: $\mathcal{R}$ is all of x:
   This models the usual Ziv-Merhav operating mode (needed in Sec. V-A), and $\mathcal{A}_\mathcal{R} = \mathcal{A}_x$;
3) $y_-|x$: $\mathcal{R}$ is the past of both x and y:
   This will be needed later on in Sec. V-B, and $\mathcal{A}_\mathcal{R} = \mathcal{A}_x \cup \mathcal{A}_y$;
4) $y_-|^+x$: $\mathcal{R}$ is the past of y and all of x:
   This will be needed later on in Sec. IV-C, and $\mathcal{A}_\mathcal{R} = \mathcal{A}_x \cup \mathcal{A}_y$;

These types of conditional informations are depicted in Fig. 1. When the conditional information is left unspecified, we will use $x \wr y$ to stand for either type of conditional information.

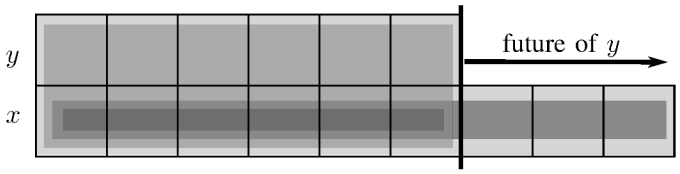

Fig. 1. Sequences $\mathcal{R}$ for the conditional information (from which references are allowed), by darkening shades of gray: $y_-|^+x$, $y_-|x$, $y|^+x$ and $y|x$. The thick vertical bar represents the position of the current lookahead buffer when encoding y.

Using any conditional information of the above, our LZ generic coder will always produce *symbols* of the form (l, v), which can be either:

- *references*: $l > 1$ is the length [1] of a subsequence in the dictionary, and, although it is not used in this paper, v is the offset in $\mathcal{R}$ at which bytes should start to be copied;
- *literals*: $l = 1$ and v is the literal in x that should be copied to the output buffer.

[1]Internally, our dictionary data structure is a three-byte indexed array of 256^3 unrolled linked lists. This trick allows us to reach subsequences of size 2: if the list indexed by the 3 bytes in the lookahead buffer is empty, then we scan for emptiness the remaining 255 slots indexed by the first two bytes in the lookahead buffer. If there are only empty lists, a literal has to be emitted (first byte of the lookahead buffer), otherwise we return immediately the length value 2 as soon as we stumble upon a non-empty list. However costly in memory (especially regarding common L2 and L3 cache sizes, and the fact that we never delete any reference in memory due to the semi-infinite sliding window), we already enjoy decent running times. Multi-threaded dictionary search shall be added shortly.

Eventually, our coder will factorize a sequence x given another, known sequence y into n symbols by finding always the longest subsequences:

$$x \wr y \rightsquigarrow (l_1, v_1) \dots (l_n, v_n)\,.$$

Let the set of lengths produced during factorization be $\mathcal{L}_{x\wr y}$.

IV. A GENERIC, LZ-BASED CONDITIONAL COMPLEXITY ESTIMATE

Besides using a compressor as in the NCD, the traditional approach to estimating a Kolmogorov complexity is to count the number of subsequences (see elsewhere the abundant literature on the LZ complexity in biomedical signal analysis). The symbol length information that we use instead, already captures much of the amount of information in x that is contained in $\mathcal{R}$ (as the results will demonstrate, it delivers much sharper estimates than the size of compressed files as in the NCD). We shall actually define a family of estimates, parameterized by a so-called *admissible* function, which can be used to tune the way the symbol length information is taken into account.

A. Definitions

Definition IV.1. *Admissible function.*

A function $f : \mathbb{N}^\star \to [0, 1]$ is said to be admissible iff it is monotonically increasing.

Definition IV.2. *Set value.*

Let $f : \mathbb{N}^\star \to \mathbb{R}$ be a mapping and let $\mathcal{S}$ be a finite set of non-zero natural numbers. The image of $\mathcal{S}$ by f is defined as:

$$|\mathcal{S}|_f = \sum_{s\in\mathcal{S}} f(s).$$

The notation $|\mathcal{S}| = |\mathcal{S}|_{\mathbf{1}_\mathcal{S}}$ will also be used to denote the cardinal of $\mathcal{S}$.

Definition IV.3. *Generic conditional complexity estimate.*

Given an admissible function f, and two non-empty sequences $x \in \mathcal{A}_x^+$ and $y \in \mathcal{A}_y^+$, our conditional complexity estimate of x given y, denoted $S_f(x|y)$, is defined as:

$$S_f(x \wr y) = \left(1 - \frac{\sum_{\mathcal{L}_{x\wr y}} l f(l) - (|\mathcal{L}_{x\wr y}|_f - 1)}{|x|}\right) \frac{|\mathcal{L}_{x\wr y}| - 1}{|x|}. \tag{5}$$

which can be factorized like: $S_f(x \wr y) = \mathcal{S}\mathcal{Z}$.

In Eq. (5), the two-terms factorization elements of $S_f(x \wr y)$ can be interpreted the following way:

1) $\mathcal{S}$ is based on the length ratio of x that is explained by y – we will show that it acts as a "spreading" factor that emphasizes differences between both sequences so that the final value allows for a sharper numerical estimate (see Sec. VI-A);
2) $\mathcal{Z}$ is the normalization of our approximation of the relative complexity [6]. This normalization is simply obtained by dividing by the maximum number of symbols that can be produced, namely $|x|$.

ISBN 978-0-9928626-7-1 © EURASIP 2017

Lemma 1. $0 \leq S_f(x \wr y) < 1$.

Proof. See [7]. □

B. Soft estimates

We start by choosing an admissible function f. The use of f in Eq. 5 allows to modulate the choice of the references taken into account, and how they contribute to the construction of the estimate of the complexity.

Any choice for f will have to take meaninfulness of references into account. Such meaninful references are first defined below (in short, meaninful references are not caused by randomness).

Definition IV.4. *Meaningful references [8].*

A reference (l, v) is said to be meaningful with respect to $\mathcal{R}$ iff:

$$l > l^0_{\mathcal{R}} = \log_{|\mathcal{A}_{\mathcal{R}}|} |\mathcal{R}| . \tag{6}$$

Among all possible choices for f, we arbitrarily favor C^∞ functions and make use of a sigmoid. The very details on why this is a reasonable choice are to be found in [7].

Definition IV.5. *Sigmoid function.*

The admissible sigmoid function for sequence $\mathcal{R}$, denoted $f^s_{\mathcal{R}}$, is defined as:

$$f^s_{\mathcal{R}}(l) = \frac{1}{1 + e^{-l+l^0_{\mathcal{R}}}}.$$

C. Self-complexity and joint complexity

Let $\mathcal{L}_x = \mathcal{L}_{x|x}$ be the set of lengths produced during a regular LZ77 factorization (*i.e.*, a close version of that in [9]).

Definition IV.6. *Self-complexity estimate.*

Given an admissible funtion f and a non-empty sequence $x \in \mathcal{A}_x^+$, the self-complexity of x, denoted $S_f(x)$, is estimated as:

$$S_f(x) = S_f(x|x).$$

This allows us to propose an estimate for the joint complexity of sequences x and y. The joint Kolmogorov complexity can be understood as the minimal program length able to encode both x and y, as well as a means to separate the two [10]. Hence, there is no need to restrict the references only to x, and we should allow references to the past of y as well. In order to mimic the relationship $K(x, x) = K(x)$, we choose a length ratio as the way to separate both sequences.

Definition IV.7. *Joint complexity estimate.*

Given an admissible function f, and two non-empty sequences $x \in \mathcal{A}_x^+$ and $y \in \mathcal{A}_y^+$, the joint complexity of x and y, denoted $S_f(x, y)$, is estimated as:

$$S_f(x, y) = S_f(y\_|^+x) + S_f(x) + \log_{|\mathcal{A}_x|}\left(\frac{|x|}{|y|}\right).$$

Note that $S_f(x, x) = S_f(x)$ because $S_f(x\_|^+x) = 0$.

In order to validate our approach, we have measured the following absolute error: $\epsilon = |S_f(x, y) - S_f(y, x)|$. We obtained a maximum average absolute error value below 2.37% (when comparing *e.g.*, DNA and human texts), this value being much lower when sequences encode the same type of data (*e.g.*, two DNA sequences), typically less than 1%.

V. NSD and directed information

In this section, we use the previous definitions to devise both an algorithmic semi-distance and directed information definitions that are key to the applications in Sec. VI.

A. The normalized semi-distance

Since $S_f(x|^+y)$ is normalized, we can now refer directly to Eq. (1) to propose a semi-distance.

Definition V.1. *NSD_f.*

Given an admissible function f, and two non-empty sequences $x \in \mathcal{A}_x^+$ and $y \in \mathcal{A}_y^+$, our normalized semi-distance, denoted NSD_f, is defined as:

$$NSD_f(x, y) = max\left\{S_f(x|^+y), S_f(y|^+x)\right\}.$$

Note that NSD_f stands for Normalized Semi-Distance using f. By default, when $f = f^s_y$, it is simply denoted by NSD.

Theorem. *NSD_f is a normalized semi-distance.*

Proof. See [7]. □

Note that using a LZ-based compressor actually makes the NCD a semi-distance [7].

B. Directed algorithmic information

Causality inference relies on the assessment of a matrix of directed informations from which a causality graph will be produced. Due to the very nature of causality, some fundamental restrictions on the underlying graph structure apply. In particular, most authors focus on directed acyclic graphs (DAG) [11] and we will hereafter follow this line. Therefore, we start by defining estimates of directed algorithmic information.

We would like to stress that causality has received several interpretations and it is, among other considerations, also dependent on the type of data at hand. We will consider two types of data here: time series [12] (for which a version based on classical information theory has been proposed [13]), and data that is not a function of time [11]. To some extent, this relates to the difference between online and offline applications. Therefore, we need to distinguish between the two.

Let $X = \{x_i\}$ be a set of sequences, and let us denote $X \backslash Y$ the set from which the set of sequences Y was removed ($Y \subset X$). When $Y = \{y\}$, we also write $X \backslash y$.

We formulate the causal directed algorithmic information as follows:

Definition V.2. *Causal directed algorithmic information.*

$$\forall i \neq j,\ C(x_i \to x_j) = K(x_j\_|X \backslash \{x_i, x_j\}) - K(x_j\_|X \backslash x_j). \tag{7}$$

ISBN 978-0-9928626-7-1 © EURASIP 2017

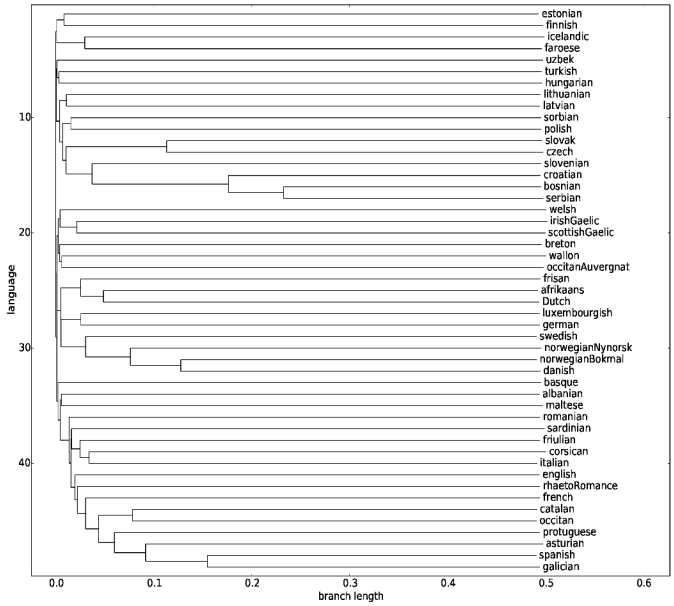

Fig. 2. Clustering ([14]) of writing systems using NCD/`gzip` All text excerpts below 16KiB to accomodate the `gzip`/`DEFLATE` sliding window size.

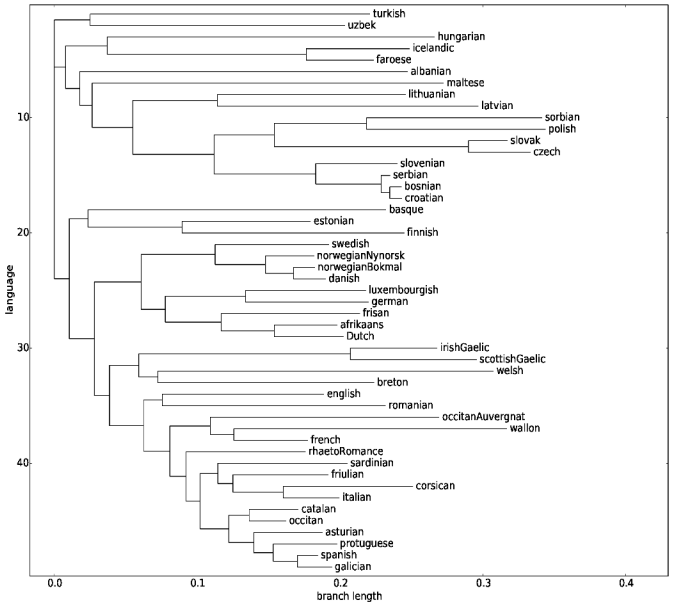

Fig. 3. Clustering ([14]) of writing systems using NSD.

$C(x_i \to x_j)$ is the amount of algorithmic information flowing from x_i to x_j when observing data online in real time (think of the x_i as *e.g.*, outputs of ECG probes).

In practice, we compute:

$$C_S(x_i \to x_j) = S(x_{j\_}|X\setminus\{x_i, x_j\}) - S(x_{j\_}|X\setminus x_j). \quad (8)$$

Similarly, for offline applications, when all the data is available beforehand (think *e.g.*, of text excerpts), we define the so-called full directed algorithmic information as:

Definition V.3. *Full directed algorithmic information.*

$$\forall i \neq j,\ F(x_i \to x_j) = K(x_{j\_}|^+X\setminus\{x_i, x_j\}) - K(x_{j\_}|^+X\setminus x_j). \quad (9)$$

In practice, we compute:

$$F_S(x_i \to x_j) = S(x_{j\_}|^+X\setminus\{x_i, x_j\}) - S(x_{j\_}|^+X\setminus x_j). \quad (10)$$

Note that we are only considering the amount of information flowing from one sequence to another. Hence, we are fundamentally fitting in the Markovian framework. And since we remove the influence of all other sequences, we are actually measuring the influence of the sole innovation contained in one such sequence onto another.

VI. Results

In this section, we present some results on real data. More results, especially on synthetic data, are available in [7].

A. Clustering languages

The effect of the first term in Eq. (5) is clear: the trees are much more airy (Fig. 3) than when using NCD/`gzip` (Fig. 2). One has obviously a real advantage in taking into account the lengths produced by the factorization.

B. An experiment in literature

Jean-Philippe Toussaint is a famous Belgian author of French expression with a specific way of writing: he works by producing paragraphs one after the other. Each paragraph gets typeset, annotated by hand, typeset again, annotated again, and so on until the author is satisfied. Some of his paragraphs culminate to more than 50 successive versions. In Fig. 4, we show the eight successive versions of one of his paragraphs (he does not necessarily typeset exactly the annotated version but makes changes in between). These versions are called fragments in Fig. 4 and Fig. 5. As one can see in top and middle plots of Fig. 5 (clustering using resp. Neighbor-Joining and UPGMA), our semi-distance allows to correctly recover the chronological order of the fragments.

On the lower graph of Fig. 5, all arrows have been kept in order to allow in-depth inspection of the amount of differential innovation. This representation is certainly richer as it allows to grasp the amount of information that has been reused from one fragment to another. All three results allow to correctly recover the order with which the fragments have been actually written by Jean-Philippe Toussaint.

VII. Perspectives

A careful reader has probably already noted that the spreading term $\mathcal{S}$ alone could be seen as leading to a semi-distance in its own right. Further work shall be devoted to (1) see whether the term $\mathcal{Z}$ could be dropped, and (2) provide un-normalized expressions for common complexities (self, joint).

References

[1] M. Li, X. Chen, X. Li, B. Ma, and P. M. Vitányi, "The Similarity Metric," *IEEE Transactions on Information Theory*, vol. 50, pp. 3250–3264, Dec. 2004. [Online]. Available: http://dx.doi.org/10.1109/tit.2004.838101

ISBN 978-0-9928626-7-1 © EURASIP 2017

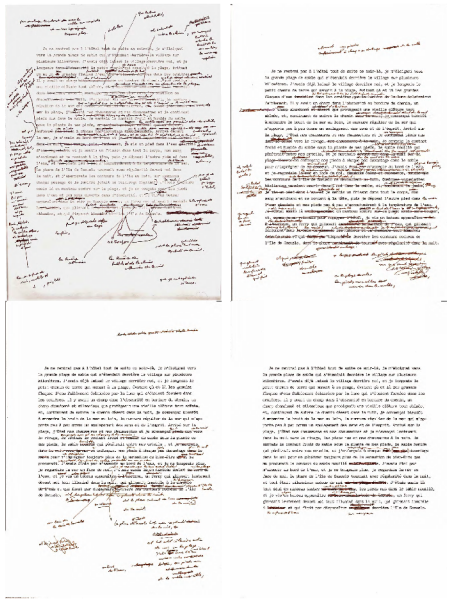

Fig. 4. Draft scans of the last pages of Jean-Philippe Toussaint's novel *La Réticence* [15], by the author (freely available at `jptoussaint.com`). The transcripts we have used are labeled as follows: Fragments 1 and 2 are the typeset and annotated versions of the upper-left scan, fragments 3 and 4 are the typeset and annotated versions of the upper-right scan, fragments 5 and 6 are the typeset and annotated versions of the lower-right scan, and fragments 7 and 8 are the typeset and annotated versions of the lower-left scan.

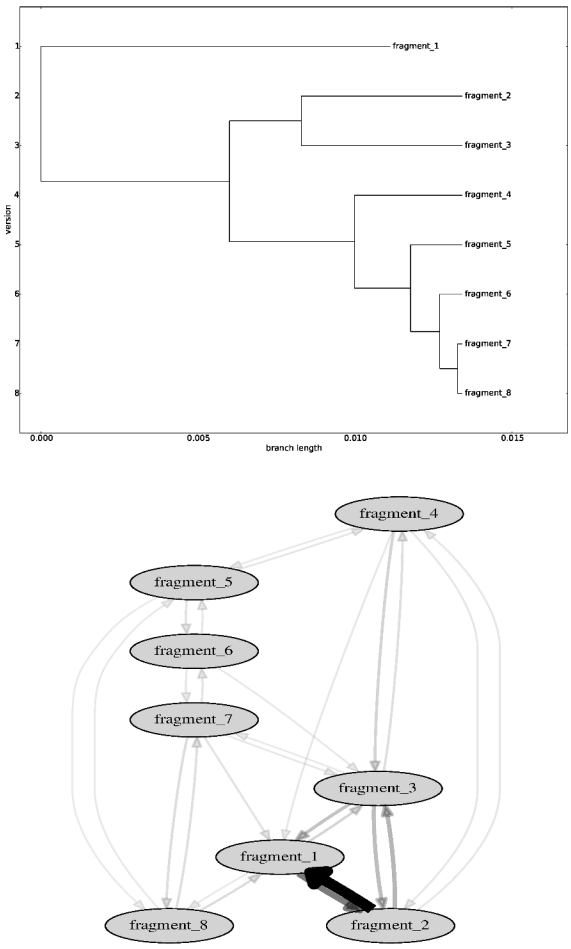

Fig. 5. Top: The distance matrix depicted using UPGMA [16] clustering. This is the method of choice one would use in this case. Bottom: The inferred causality graph using F_S (because the author may have – actually, has – moved parts of the fragments later in the text).

[2] C. H. Bennett, P. Gacs, M. Li, P. M. Vitányi, and W. H. Zurek, "Information Distance," *IEEE Transactions on Information Theory*, vol. 44, pp. 1407–1423, Jul. 1998. [Online]. Available: http://dx.doi.org/10.1109/18.681318

[3] L. P. Deutsch, "DEFLATE Compressed Data Format Specification, version 1.3," 1996. [Online]. Available: http://www.gzip.org/zlib/rfc-deflate.html

[4] I. Pavlov, "7z Format."

[5] M. Cebrián, M. Alfonseca, and A. Ortega, "Common Pitfalls Using the Normalized Compression Distance: What to Watch Out for in a Compressor," *Communications in Information and Systems*, vol. 5, pp. 367–384, 2005. [Online]. Available: http://dx.doi.org/10.4310/cis.2005.v5.n4.a1

[6] J. Ziv and N. Merhav, "A Measure of Relative Entropy Between Individual Sequences with Application to Universal Classification," *IEEE Transactions on Information Theory*, vol. 39, pp. 1270–1279, Jul. 1993. [Online]. Available: http://dx.doi.org/10.1109/isit.1993.748668

[7] M. Revolle, F. Cayre, and N. Le Bihan, "SALZA: Soft Algorithmic Complexity Estimates for Clustering and Causality Inference," *DRAFT*, 2016. [Online]. Available: http://arxiv.org/pdf/1607.05144v1.pdf

[8] A. Lempel and J. Ziv, "On the Complexity of Finite Sequences," *IEEE Transactions on Information Theory*, vol. 22, pp. 75–81, January 1976. [Online]. Available: http://dx.doi.org/10.1109/tit.1976.1055501

[9] J. Ziv and A. Lempel, "A Universal Algorithm for Sequential Data Compression," *IEEE Transactions on Information Theory*, vol. 23, pp. 337–343, May 1977. [Online]. Available: http://dx.doi.org/10.1109/tit.1977.1055714

[10] M. Li and P. M. Vitányi, *An Introduction to Kolmogorov Complexity and Its Applications*. Springer-Verlag New-York, 2008. [Online]. Available: http://dx.doi.org/10.1007/978-0-387-49820-1

[11] J. Pearl, *Causality, Models, Reasoning, and Inference*. Cambridge University Press, 2009. [Online]. Available: http://dx.doi.org/10.1017/cbo9780511803161

[12] C. W. Granger, "Investigating Causal Relations by Econometric Models and Cross-spectral Methods," *Econometrica*, vol. 37, pp. 424–438, Aug. 1969. [Online]. Available: https://dx.doi.org/10.2307/1912791

[13] P.-O. Amblard and O. Michel, "The Relation between Granger Causality and Directed Information Theory: A Review," *Entropy*, vol. 15, pp. 113–143, 2013. [Online]. Available: http://dx.doi.org/10.3390/e15010113

[14] N. Saitou and M. Nei, "The Neighbor-Joining Method: A New Method for Reconstructing Phylogenic Trees," *Molecular Biology and Evolution*, vol. 4, pp. 406–425, July 1987.

[15] J.-P. Toussaint, *La Réticence*. Les Éditions de Minuit, 1991. [Online]. Available: http://www.leseditionsdeminuit.fr/livre-La_R\%C3\%A9ticence-1878-1-1-0-1.html

[16] R. Sokal and C. Michener, "A Statistical Method for Evaluating Systematic Relationships," *The University of Kansas Scientific Bulletin*, vol. 38, pp. 1409–1438, March 1958.

ISBN 978-0-9928626-7-1 © EURASIP 2017