Correlation of Automorphism Group Size and Topological Properties with Program-size Complexity Evaluations of Graphs and Complex Networks

Hector Zenil[1,3,*], Fernando Soler-Toscano[2,3], Kamaludin Dingle[4,5], and Ard A. Louis[4]

[1]Unit of Computational Medicine, Karolinska Institute, Sweden.
[2]Grupo de Lógica, Lenguaje e Información, Universidad de Sevilla, Spain.
[3]Algorithmic Nature Group, LABoRES, Paris, France.
[4]Rudolf Peierls Centre for Theoretical Physics, University of Oxford, UK.
[5]The Systems Biology DTC, University of Oxford, UK.
(Dated: February 25, 2014)

We show that numerical approximations of Kolmogorov complexity (K) of graphs and networks capture some group-theoretic and topological properties of empirical networks, ranging from metabolic to social networks, and of small synthetic networks that we have produced. That K and the size of the group of automorphisms of a graph are correlated opens up interesting connections to problems in computational geometry, and thus connects several measures and concepts from complexity science. We derive these results via two different Kolmogorov complexity approximation methods applied to the adjacency matrices of the graphs and networks. The methods used are the traditional lossless compression approach to Kolmogorov complexity, and a normalised version of a Block Decomposition Method (BDM) based on algorithmic probability theory.

Keywords: Kolmogorov complexity; graph automorphism; complex networks; graph automorphisms; algorithmic probability; Block Decomposition Method; compressibility; biological networks; network biology.

I. INTRODUCTION

Graphs are an important tool for mathematically analysing many systems, from interactions of chemical agents, to ecological networks, to representing data objects in computer science [1, 2]. An interesting perspective regarding such graphs is to investigate the complexity [3, 4] or information content of a graph [5]. While Shannon information theory [5–7] and counting symmetries [8, 9] have been applied to measure information content/complexity of graphs, little has been done, by contrast, to demonstrate the utility of Kolmogorov complexity as a numerical tool for graph and real-world network investigations. Some theoretical connections between graphs and algorithmic randomness have been explored (e.g. [10]), but these are mostly related to formal properties of random graphs.

Here we computationally study some of these numerical and real-world directions and show how Kolmogorov complexity can capture group-theoretic and topological properties of abstract and empirical graphs and networks. We do this by introducing a measure of graph complexity based on approximating the Kolmogorov complexity of the adjacency matrix representation of a graph, which we achieve by applying our recently developed *Block Decomposition Method* (below) [11]. A theoretical advantage of using Kolmogorov complexity K is that the measure K is designed to capture *all* structure (i.e. non-randomness) in an object, such as a graph. In contrast, only looking at symmetries, for example, can miss structure and potential simplicity in a graph. For example, Fig. 1 shows a graph with no symmetries, despite being far from random, and indeed is intuitively simple.

This paper is structured as follows: The next two Sections give background definitions and theorems; then in Section IV we consider the question of whether approximations of Kolmogorov complexity can characterise group-theoretic and topological properties of graphs and networks, more specifically, we study the number of automorphisms of a graph and introduce a normalised measure of graph complexity; following this in Section V we apply our measure to real-world networks; and finally we study in Section VI algorithmic information-theoretic similarities of networks with similar topologies created using different mechanisms (e.g. random versus preferential attachment).

II. PRELIMINARIES

A. Graph notation

A graph $g = (V, E)$ consists of a set of vertices V (also called nodes) and a set of edges E. Two vertices, i and j, form an edge of the graph if $(i, j) \in E$. Let the binary adjacency matrix of g be denoted by $Adj(A)$. A graph can be represented by its adjacency matrix. Assuming that the vertices are indices from 1 to n, that is, that $V = \{1, 2, \ldots, n\}$, then the adjacency matrix of g is an $n \times n$ matrix, with entries $a_{i,j} = 1$ if $(i, j) \in E$ and 0 otherwise. The distance $D(g)$ of a graph g is the maximum distance between any 2 nodes of g. The size $V(g)$ of a graph g is the vertex count of g; similarly $E(g)$ denotes the edge count of g.

* hectorz@labores.eu

Definition 1. Two graphs g and h are *isomorphic* if and only if there exists a permutation λ such that $\lambda(g) = h$. (That is, g and h are topologically equivalent).

FIG. 1. An example of a graph which has no symmetries (i.e. automorphism group size of 1, cf. Def. 2), despite being intuitively simple, i.e. just a string of nodes with a single side node.

The general problem of graph isomorphism appears, for example, in chemistry [12, 13] and biology [14, 15].

Definition 2. An *automorphism* of a graph g is a permutation λ of the vertex set V, such that the pair of vertices (i, j) forms an edge if and only if the pair $(\lambda(i), \lambda(j))$ also forms an edge.

The set of all automorphisms of an object forms a group, called the *automorphism group*. Intuitively, the size of the automorphism group $A(g)$ provides a direct measure of the abundance of symmetries in a graph or network. Every graph has a trivial symmetry (the identity) that maps each vertex to itself.

A clustering coefficient is a measure of the degree to which nodes in a graph tend to cluster together (for example, friends in social networks [16]).

Definition 3.

$$C(v_i) = \frac{2\,|E(N_i)|}{n_i(n_i - 1)}$$

where $E(N_i)$ denotes the set of edges with both nodes in N_i, and $C(v_i)$ is the local clustering coefficient of node v_i.

III. APPROXIMATING KOLMOGOROV COMPLEXITY

A. Background Results

Before describing our graph complexity measure, we provide some pertinent definitions and results from the theory of algorithmic randomness.

The *Kolmogorov complexity* of a string s is the length of the shortest program p that outputs the string s, when run on a universal Turing machine U. Formally [17, 18],

Definition 4.

$$K_U(s) = \min\{|p|, U(p) = s\} \tag{1}$$

By the *invariance theorem* [19], K_U only depends on U up to a constant c, so as is conventional, the subscript U of K is dropped. A technical inconvenience of K as a function taking s to the length of the shortest program that produces s, is that K is semi-computable. Lossless compression algorithms are, however, traditionally used to approximate K given that compressibility is a sufficient test of low Kolmogorov complexity.

The concept of algorithmic probability (also known as Levin's semi-measure) yields a method to approximate Kolmogorov complexity. The algorithmic probability of a string s is a measure that describes the probability that a random program p produces the string s when run on a universal (prefix-free [20]) Turing machine U. Formally [18, 21, 22],

Definition 5.

$$m(s) = \sum_{p:U(p)=s} 1/2^{|p|} \tag{2}$$

The probability semi-measure $m(s)$ is related to Kolmogorov complexity $K(s)$ in that $m(s)$ is at least the maximum term in the summation of programs $(m(s) \geq 2^{-K(s)})$, given that the shortest program carries the greatest weight in the sum. The algorithmic Coding Theorem [23, 24] further establishes the connection between $m(s)$ and $K(s)$.

Theorem 1. ([22]):

$$|-\log_2 m(s) - K(s)| < c \tag{3}$$

where c is some fixed constant, independent of s. The theorem implies [23] (pp. 190-191) and [25, 26] that one can estimate the Kolmogorov complexity of a string from the frequency of production from running random programs by simply rewriting Eq. (3) as:

$$K(s) = -\log_2 m(s) + O(1) \tag{4}$$

The advantage of calculating $m(s)$ as an approximation of K by application of the Coding Theorem is that $m(s)$ retrieves finer-grained values than compression algorithms, which are unable to compress and therefore distinguish complexity variations for small objects. Applications of $m(s)$ have been explored in [25, 27–31], and include applications to image classification [11] and to the evolution of space-time diagrams of cellular automata [32]. Error estimations of approximations of $m(s)$ have also been reported before in [11], showing that estimations of the constant involved in the invariance theorem remain small when taking larger samples of Turing machines from which $m(s)$ is calculated. Moreover, in the same paper [11], it is shown that values between $m(s)$ and lossless compression algorithms, for experiments where both methods overlap, remain close to each other. Here Fig. 5 provides an indication of the error estimation as

compared to the estimation error of the lossless compression algorithm used (Deflate), by comparing the Kolmogorov complexity of dual graphs estimated using both $m(s)$ and lossless compression.

B.   The Coding Theorem Method

In [28] a technique was advanced for approximating $m(s)$ (hence K) by means of a function that considers all Turing machines of increasing size (by number of states). Let (n,k) denote the set of Turing machines with n states and k symbols using the Busy Beaver formalism [33] and let T be a Turing machine in (n,k) with empty input. Then:

Definition 6.

$$\mathbb{D}(n,k)(s) = \frac{|\{T \in (n,k) : T \ produces \ s\}|}{|\{T \in (n,k) : T \ halts \ \}|} \quad (5)$$

For small values n and k, $\mathbb{D}(n,k)$ is computable for values of the Busy Beaver problem that are known. The Busy Beaver problem is the problem of finding the n-state, k-symbol (or colour) Turing machine which writes a maximum number of non-blank symbols on the tape before halting and starting from an empty tape, as originally proposed by Rado [33]. Or alternatively, finding the Turing machine that performs a maximum number of steps when started on an initially blank tape before halting. For $n = 4$ and $k = 2$, for example, the Busy Beaver machine has maximum runtime $S(n) = 107$ [34], from which one can deduce that if a Turing machine with 4 states and 2 symbols running on a blank tape hasn't halted after 107 steps then it will never stop. This is how $\mathbb{D}$ was initially calculated with the help of the Busy Beaver. However, the Busy Beaver problem is only computable for small n, k values. Nevertheless, one can continue approximating $\mathbb{D}$ for a greater number of states (and colours) proceeding by sampling as described in [30], with an informed runtime after studying the behaviour of the runtime distribution of the Busy Beavers.

The Coding Theorem Method [28, 30] is rooted in the relation provided by algorithmic probability between frequency of production of a string from a random program and its Kolmogorov complexity (Eq. (3)). Essentially it uses the fact that the more frequent a string is, the lower Kolmogorov complexity it has; and strings of lower frequency have higher Kolmogorov complexity.

C.   Applying the Block Decomposition Method to graphs

As an extension of the Coding Theorem Method [28, 30] of approximating bit string complexity (above), a method to approximate the Kolmogorov complexity of d-dimensional objects was advanced in [11]. The method is called the *Block Decomposition Method* (BDM) and it consists of decomposing larger objects into smaller pieces for which complexity values have been estimated, then reconstructing an approximation of the Kolmogorov complexity of the larger object by adding the complexity of the individual pieces according to rules of information theory.

By taking the adjacency matrix of a graph as the representation of the graph, we can use the BDM to estimate K for graphs. Given the 2-dimensional nature of adjacency matrices we can use a variation of a Turing machine that runs on a 2-dimensional tape in order to estimate upper bounds of K of the adjacency matrix of a graph. A popular example of a 2-dimensional tape Turing machine is Langton's ant [35]. Another way to see this approach is to take the BDM as a form of deploying all possible 2-dimensional deterministic Turing machines of a small size in order to reconstruct the adjacency matrix of a graph from scratch (or smaller pieces that fully reconstruct it). Then as with the Coding Theorem Method (above), the Kolmogorov complexity of the adjacency matrix of the graph can be estimated via the frequency that it is produced from running random programs on the (prefix-free) 2-dimensional Turing machine.

Having outlined the theory, here we will use Eq. (5) with $n = 5$ and $k = 2$ to approximate K. The choice of this n and k is because 2-dimensional Turing machine with $n = 4$ states and empty input produce all square arrays (or matrices) of size 3 by 3 but not all of size 4 by 4. The larger the arrays, the better approximations of K for large objects decomposed in square arrays, and $n = 5$ was a Turing machine size that we were able to run with current technology on a medium-size supercomputer ($\sim$ 30 cpus) for about a month. Details are provided in [30]. One can continue calculating further and approximations will improve accordingly.

Let $\mathbb{D}(5,2)$ be the frequency distribution constructed from running 2-dimensional machines (hence producing arrays rather than strings) according to Eq. (5). Then, for an array s,

Definition 7.

$$K_m(s) = -\log_2(\mathbb{D}(5,2)(s)) \quad (6)$$

where $K_m(s)$ is an approximation to K by means of the Coding Theorem. $K_m(s)$ will then be used to estimate the complexity of graphs as follows: Formally, we will say that a graph g has complexity:

Definition 8.

$$K\log_m(g) = \sum_{(r_u,n_u)\in Adj(g)_{d\times d}} \log_2(n_u) + K_m(r_u) \quad (7)$$

where $Adj(g)_{d\times d}$ represents the set with elements (r_u, n_u), obtained when decomposing the adjacency matrix of g into non-overlapping squares of size d by d. In each (r_u, n_u) pair, r_u is one such square and n_u its multiplicity (number of occurrences). From now on both

K_m and $K \log_m$ will denote the same Eq. (7) given that Eq. (6) will no longer be used.

In [11] a set of 2-dimensional Turing machines was executed to produce all square arrays of size $d = 4$. This is why BDM is needed in order to decompose objects of larger size into objects for which its Kolmogorov complexity has been estimated. Using squares of size 4 is like looking for local regularities that give an approximation of the Kolmogorov complexity of the larger object. Because we will take $d = 4$ from this point on we will denote by K_m (sometimes also by $K \log_m$, but m should not be taken as the base of the log, which is 2 in the definition of $K \log_m$). The reader should simply bear in mind that K_m and $K \log_m$ are approximations of Kolmogorov complexity using the Block Decomposition Method (BDM).
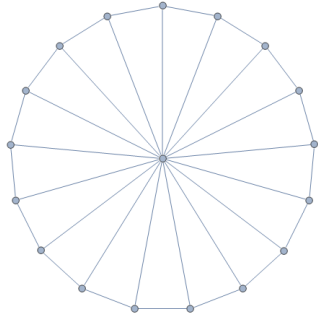
D. Complexity and graph vertex order
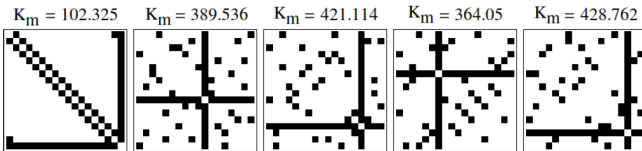

FIG. 2. Wheel-18 graph.

FIG. 3. Wheel-18 graph with the nodes in five different orders. Despite each adjacency matrix corresponding to the same topology (i.e. the Wheel-18 graphs), the different node orders affect the estimated complexity.

As a final note on our method of approximating K, the order of the graph nodes in the adjacency matrix is relevant for the complexity retrieved by BDM. This is especially important in highly symmetrical graphs. Fig. 3 shows five different adjacency matrices for the wheel graph of 18 nodes (Fig. 2), with the nodes ordered in different ways. The extreme left matrix of Fig. 3 represents the nodes in consecutive order, with the central one at the right. The other four matrices sort the nodes in random order. As we can see in this very regular graph, the lowest complexity corresponds to the organized matrix.

Hence, when studying the complexity of graphs, for several applications we will be interested not only in the K_m value of a particular adjacency matrix, but in several randomisations of the vertex order. Notice that it is just the adjacency matrix representation that changes; topologically the graph is the same in all permutations of the nodes. In estimating complexity, it is reasonable to consider that the complexity of a graph corresponds to the lowest K_m value of all permutations of the adjacency matrix, as the shortest program generating the simplest adjacency matrix is the shortest program generating the graph. Hence when estimating K_m we find a large number of programs producing the array from which a minimum complexity value is calculated by means of the frequency of the programs producing the array.

E. Normalising BDM

We now also introduce a normalised version of the BDM. The chief advantage of a normalised measure is that it enables a comparison among objects of different sizes without allowing the size to dominate the measure. This will be useful to compare graphs and networks of different sizes. First, for a square array of size $n \times n$, we define:

$$MinBDM(n)_{d \times d} = \lfloor n/d \rfloor + \min_{x \in M_d(\{0,1\})} K_m(x)$$

Where $M_d(\{0,1\})$ is the set of binary matrices of size $d \times d$. For any n, $MinBDM(n)_{d \times d}$ returns the minimum value of Eq. (7) for square matrices of size n, so it is the minimum BDM value for graphs with n nodes. It corresponds to an adjacency matrix composed by repetitions of the least complex $d \times d$ square. It is the empty (or complete) graph, because $0_{d,d}$ and $1_{d,d}$ are the least complex squares (the most compressible) of size d.

Secondly, for the maximum complexity, Eq. ((7)) returns the highest value when the result of dividing the adjacency matrix into $d \times d$ squares contains the highest possible number of different squares (to increase the sum of the right terms in Eq. (7)) and the repetitions (if necessary) are homogeneously distributed along those squares (to increase the sum of the left terms in Eq. (7)), which should be the most complex ones in $M_d(\{0,1\})$. For $n, d \in \mathbb{N}$, we define a function

$$f_{n,d} : M_d(\{0,1\}) \longmapsto \mathbb{N}$$

that verifies:

$$\sum_{r \in M_d(\{0,1\})} f_{n,d}(r) = \lfloor n/d \rfloor^2 \quad (8)$$

$$\max_{r \in M_d(\{0,1\})} f_{n,d}(r) \leq 1 + \min_{r \in M_d(\{0,1\})} f_{n,d}(r) \quad (9)$$

$$K_m(r_i) > K_m(r_j) \implies f_{n,d}(r_i) \geq f_{n,d}(r_j) \quad (10)$$

The value $f_{n,d}(r)$ indicates the number of occurrences of $r \in M_d(\{0,1\})$ in the decomposition into $d \times d$ squares

of the most complex square array of size $n \times n$. Condition Eq. (8) establishes that the total number of component squares is $\lfloor n/d \rfloor^2$. Condition Eq. (9) reduces the square repetitions as much as possible, to increase the number of differently composed squares as far as possible and distribute them homogeneously. Finally, Eq. (10) ensures that the most complex squares are the best represented. Then, we define:

$$MaxBDM(n)_{d \times d} = \sum_{\substack{r \in M_d(\{0,1\}), \\ f_{n,d}(r) > 0}} \log_2(f_{n,d}(r)) + K_m(r)$$

Finally, the normalised BDM value of a graph g is:

Definition 9. Given graph g with n nodes, $NBDM(g)_d$ is defined as

$$\frac{K_m(g) - MinBDM(n)_{d \times d}}{MaxBDM(n)_{d \times d} - MinBDM(n)_{d \times d}} \quad (11)$$

This way we take the complexity of a graph g to have a normalised value which is not dependent on the size $V(g)$ of the graph but rather on the relative complexity of g with respect to other graphs of the same size. The use of $MinBDM(n)_{d \times d}$ in the normalisation is relevant. Note that the growth of $MinBDM(n)_{d \times d}$ is linear with n, and the growth of $MaxBDM(n)_{d \times d}$ exponential. This makes that for complex graphs, the result of normalising using just $K_m(g)/MaxBDM(n)_{d \times d}$ would be similar to $NBDM(g)_d$. But it would not work for simple graphs, as when the complexity of g is close to the minimum, the value of $K_m(g)/MaxBDM(n)_{d \times d}$ drops exponentially with n. For example, the normalised complexity of an empty graph would drop exponentially in its size. To avoid it, Eq. (11) considers not only the maximum but also the minimum.

Notice the heuristic character of $f_{n,d}$. It is designed to ensure a quick computation of $MaxBDM(n)_{d \times d}$, and the distribution of complexities of squares of size $d \in \{3, 4\}$ in $\mathbb{D}(5, 2)$ ensures that $MaxBDM(n)_{d \times d}$ is actually the maximum complexity of a square matrix of size n, but for other distributions it could work in a different way. For example, condition (9) assumes that the complexities of the elements in $M_d(\{0, 1\})$ are similar. This is the case for $d \in \{3, 4\}$ in $\mathbb{D}(5, 2)$, but it may not be true for other distributions. But at any rate it offers a way of comparing the complexities of different graphs independent of their size.

Having defined our complexity measure and its normalised version, we verify it by showing that it behaves in accordance with theory when analysing edge density and dual graphs.

F. Complexity and edge density

There has been a protracted discussion in the literature as to whether the complexity of a structure increases with its connectivity, beginning with a disconnected graph with no edges, or whether instead it reaches a maximum before returning to zero for complete graphs [3]. In [36], for example, Gell-Mann asks about the algorithmic complexity (description length) of small graphs with eight vertices each and an increasing number of edges $E(g) = 0$ to $E(g) = V(g)(V(g) - 1)/2$ (complete graph). Gell-Mann reasonably argues that these two extreme cases should have roughly equal complexity. Graphs with 0.5 edge density should fall in between and the other cases are more difficult to tell apart. Here we provide an answer to the general question of the relation between description complexity and edge count, an answer which is in agreement with Gell-Mann while being at odds with several other tailor made measures of complexity [3], [37], [38] and [39]. We have created a number of random graphs, all with 50 nodes but different numbers of edges, ranging from 1 to $\binom{50}{2} = 1225$, in intervals of 1225/20 edges. For each interval, we created 20 random graphs and generated 100 random permutations of each (see Fig. 4). All the graphs (Fig. 4) in the same interval correspond to aligned points. There are always 20 aligned points, but in most cases they overlap. Points represent the minimum (Top) and standard deviations (Bottom) of the complexity of the 100 permutations of each group.

In fact, that the most complex binary strings (hence binary adjacency matrices) will have roughly equal zeros and ones is a standard result from Kolmogorov complexity theory. This can be shown straightforwardly by the following relation of K to Shannon entropy: If $x = x_1, \ldots, x_l$ is a bit string, then [23],

$$K(x) \leq lH_0 \left(\frac{1}{l} \sum_{i=1}^{l} x_i \right) + O(\log_2(l)) \quad (12)$$

where

$$H_0(p) = -p \log_2(p) - (1 - p) \log_2(1 - p) \quad (13)$$

is the Shannon entropy (in bits) of the string x. This then implies that if the number of edges diverges from roughly half of the maximum possible, then p must diverge from 0.5, and so the entropy H_0 decreases, implying that x cannot be algorithmically random. Hence, the most complex strings must have roughly equal zeros and ones, or in graph terms, they must have roughly half the number of possible edges. Additionally, Eq. (12) predicts that Fig. 4 (Top) should look roughly like the graph of $H_0(p)$ vs p, which it does: peaking at the centre, with gradual decay to zero at $p = 0, 1$. Also, note the clear symmetry of the plots in Fig. 4. This is because the value of K_m is the same for any given graph and its complement. Complexity is minimal for empty or complete graphs (the most

homogeneous matrices, all 0 or all 1), and so the standard deviation is also minimal. These observations show that our measure is behaving as expected from theory.
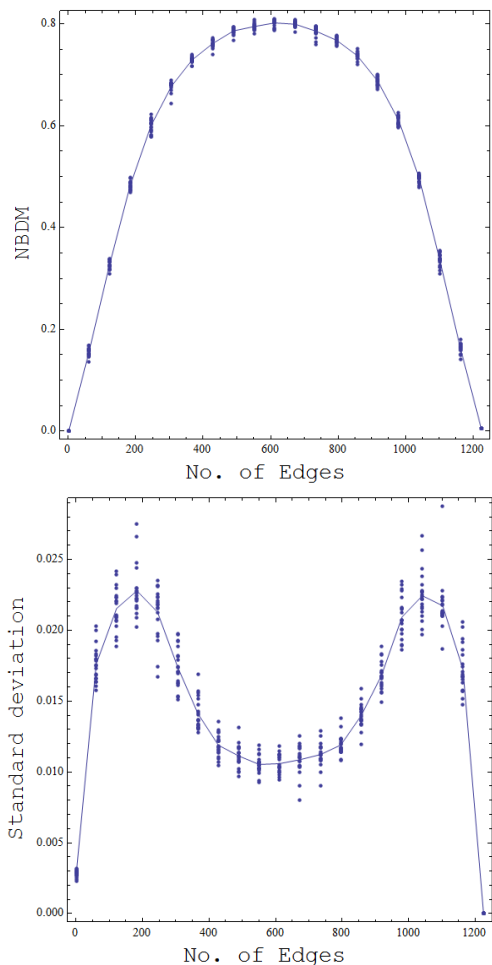

FIG. 4. Estimated (normalised) Kolmogorov complexity for increasing number of edges for random graphs of 50 nodes each. The minimum complexity (Top) and standard deviation (Bottom) is shown for 100 random permutations of 20 graphs in each group.

G. Graph duality

The dual graph of a planar graph g is a graph that has a vertex corresponding to each face of g, and an edge joining two neighbouring faces for each edge in g. If $g\prime$ is a dual graph of g, then $A(g\prime) = A(g)$, making the calculation of the Kolmogorov complexity of graphs and their dual graphs interesting because of the correlation between Kolmogorov complexity and $A(g\prime)$, which should be the same for $A(g)$. One should also expect the estimated complexity values of graphs to be the same as those of their dual graphs, because the description length of the dual graph generating program is $O(1)$.

Note that unlike the t-statistic, the value of the D statistic (and hence the P value) is not affected by scale changes like using log. The KS-test is a robust test that cares only about the relative distribution of the data.

We numerically approximated the Kolmogorov complexity of dual graphs using both lossless compression and the BDM applied to the adjacency matrices of 113 regular graphs with non-identical dual graphs found in *Mathematica*'s built-in repository GraphData[]. The values (see Figs. 5) were normalised by a multiple of the size of the adjacency matrices $c|Adj(A)|$. Graphs (g) and their dual graphs ($g\prime$) were found to have estimated Kolmogorov complexity values that are close to each other. The Spearman coefficient r between $K_m(g)$ and $K_m(g\prime)$ NBDM estimations amounts to $r = 0.96$. Values approximated by lossless compression were calculated with Deflate (a standard compression algorithm) implemented in *Mathematica*'s Compress[] function. Notice that the BDM (and therefore the NBDM) accumulates errors for larger graphs if the complementary algorithm (CTM) is not run for a greater number of Turing machines. A more accurate analysis of divergence rates and accumulated errors should be investigated. Robustness of algorithmic probability approximations for different (Turing-universal) computational formalisms (e.g. deterministic cellular automata, Post tag systems, etc.) was investigated in [29], where frequency distributions were calculated and their ranking order correlation also quantified with the Spearman coefficient.

These results show that Kolmogorov complexity applied to adjacency matrices of graphs as a measure of complexity behaves as expected, agreeing both with theory and intuition. Having verified that our measure behaves as expected, we can now apply our measure to analysing automorphism in graphs.

IV. GRAPH AUTOMORPHISMS AND KOLMOGOROV COMPLEXITY

Intuition tells us that a graph with more symmetries can be more compactly described and should therefore have a lower Kolmogorov complexity. For example, if an object is symmetrical a single symbol can be used for each repetition, in addition to an additive constant of information describing the type of transformation (e.g. rotation, reversion, translation). Specifically, any collection of graph nodes within the same orbit of a group transformation will require only the bonds of one node to be specified, with the other nodes then using the same information to specify their bonds. Hence, the number of bits required to specify the graph would be signifcantly lowered, reducing the graph Kolmogorov complexity. Consequently, as the size of its automorphism group $A(g)$ of a graph g measures the extent of symmetries in g, one would expect to find Kolmogorov complexity to be related to $A(g)$.

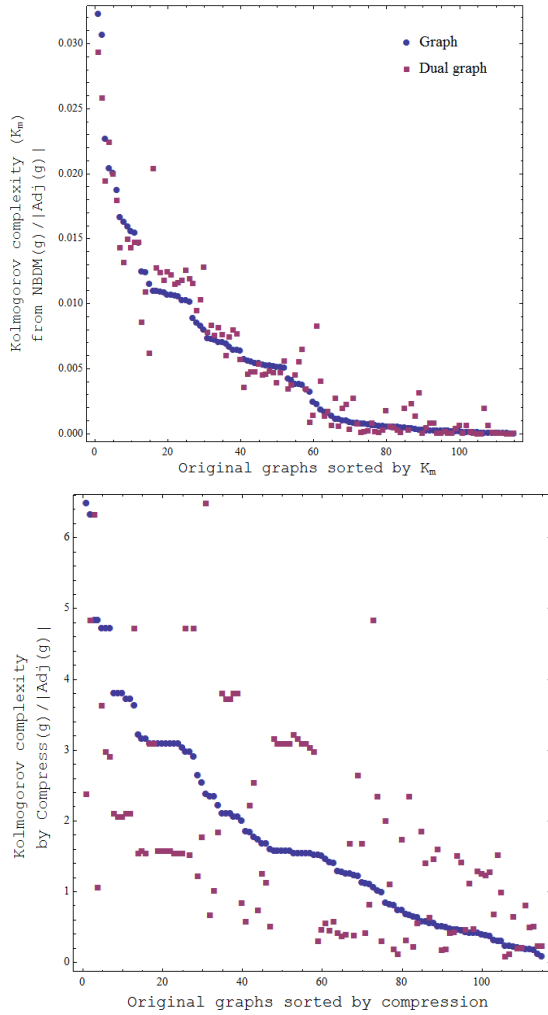We test this reasoning on connected regular graphs of

FIG. 5. Log scatter-plots of graphs ranked by Kolmogorov complexity approximated by two different methods. Top: Dual graphs not ranked by the BDM method closely follow the distribution of their respective duals as one would expect from a complexity measure given that an $O(1)$ program can build a graph from its dual and vice-versa. Bottom: Lossless compression in agreement with BDM and the expected complexity of dual graphs.
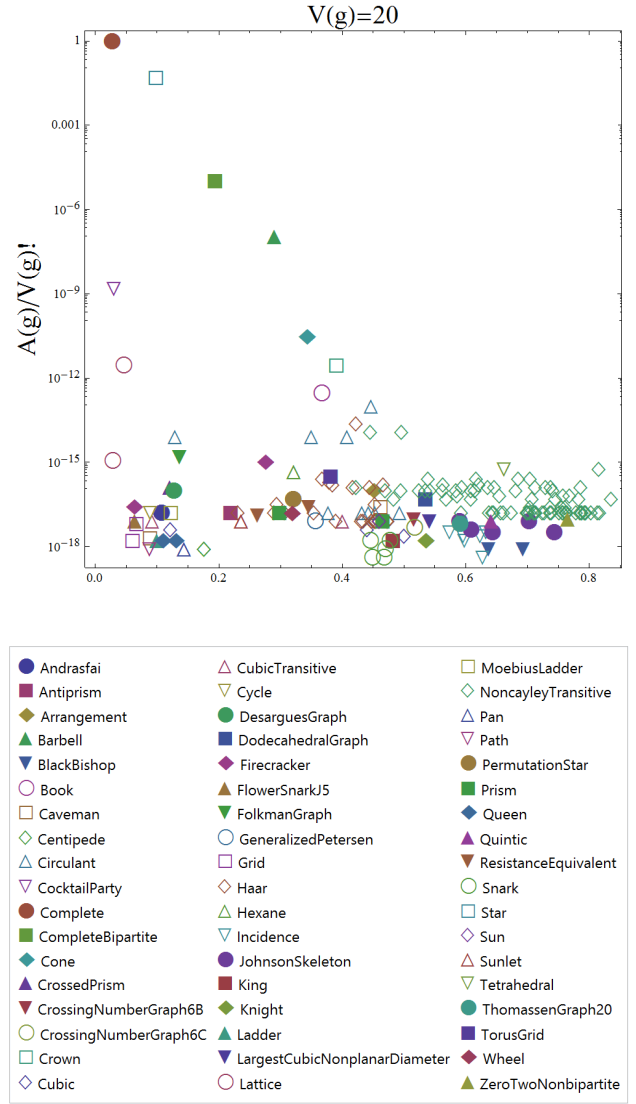

FIG. 6. Plot of graph automorphism group size $A(g)$ (y-axis) of all connected regular graphs of size $V(g) = 20$ available in *Mathematica*'s GraphData[] versus Kolmogorov complexity (x-axis) estimated by BDM. As theoretically expected, the larger automorphism group size $A(g)$ the smaller Kolmogorov complexity estimations.

size $V(g) = 20$ nodes; Fig. 6 shows that $A(g)$ and $K_m(g)$ are indeed negatively related, as expected.

It is interesting that there are several graphs of low K_m and also low $A(g)$ (see Fig. 7)—that is, there are several graphs which have few symmetries but yet also low complexity (this is analogous to Fig. 1). Hence our measure is picking up structure in the graphs which are not detectable by symmetry search.

We plotted analogous plots to Fig. 6 (i.e. connected regular graphs) for different number of nodes, using $V(g)$ between 22 and 36 (see Appendix Fig. 14 and Fig. 15). The results were qualitatively the same, with graphs of larger K estimations having smaller $A(g)$ values, and the results also agreed with those using lossless compression

(Deflate) instead of BDM.

Notice that the measure K_m does not quantitatively agree with the theoretical K, as the theoretical K has an upper bound of ~ 200 bits, which would be arrived at by specifying $Adj(g)$ literally and in full, i.e. using the fact that

$$K(g) \leq \binom{V(g)}{2} + 2\log_2\binom{V(g)}{2} + O(1) \qquad (14)$$

On the other hand, BDM retrieves values up to ~ 800. Nonetheless, BDM values (just like compression results) are consistent upper bounds. BDM can provide better approximations but it requires the calculation of a larger sample of random programs [30] with square matrices of
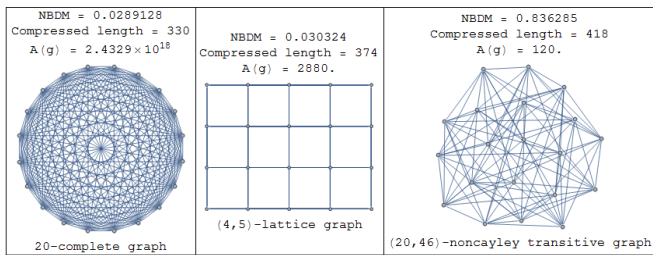
FIG. 7. The three graphs found in the boundaries of Fig. 6. From left to right: The graph at the top left, with low K_m and large automorphism group size is the complete graph for $V(g) = 20$. Bottom left with low K_m but small automorphism group size is the (4,5)-lattice. Bottom right, with high K_m and small automorphism group size: the (20,46)-noncayley transitive graph.

larger size d, compared to the current $d = 4$ that the experiments here introduced used. But in order for BDM to *scale up* and provide better K_m approximations to the theoretical (and ultimately uncomputable) K, the full method to consider is both Coding Theorem Method (CTM) + BDM. CTM is, however, computationally very expensive, while BDM is computationally very cheap, so there is a trade-off in the application of the algorithm. BDM alone is limited by the data generated by CTM, and the range of application of BDM will be limited in this respect, specially for increasingly larger objects (where lossless compression can take over and behave better than for small objects, hence CTM + BDM and lossless compression are complementary tools). The chief advantage of the CTM + BDM approach, is that CTM needs to run only once and BDM can be then efficiently applied and used many times on all sorts and types of data.

V. APPLYING BDM TO REAL-WORLD NATURAL AND SOCIAL NETWORKS

We now move on to applying the normalised version of the BDM (NBDM) to real world networks of different sizes. The 88 real-world networks range from metabolic to social networks and have between 200 and 1000 nodes, and were extracted from the function ExampleData["NetworkGraph"] in Wolfram *Mathematica* v.9. A subset of 20 of the specific real-world network examples used in this experiment are in Table I (Appendix). Fig. 8 shows complexity plotted against automorphism group size for these networks. We find that the same qualitative relationship between K and $A(g)$ as reported in Fig. 6 and Fig. 14 for synthetic networks (see Appendix) are obtained for these larger real-world networks. This provides further evidence that approximations of Kolmogorov complexity identify group-theoretic properties (as well as topological properties, as will be shown in Section VI) of graphs and networks. The automorphism group sizes $A(g)$ were calculated using the software Saucy

3.0 (http://vlsicad.eecs.umich.edu/BK/SAUCY/, accessed in September 2013), the most scalable symmetry-finding tool available today [40]. Saucy only deals with undirected graphs hence only directed versions of the real-world sample of networks was used to calculate the automorphism group size $A(g)$ of each with Saucy. We made directed graphs into undirected graphs by setting an edge between two nodes i and j if in the original directed graph there was a directed edge from i to j, j to i, or both. Clearly undirected graphs are simpler than directed graphs, in general. In the case of labelled nodes, it is easy to see that the Kolmogorov complexity of a random undirected graph is typically half of a directed labelled graph, as the binary adjacency matrix for undirected graphs is symmetric. For unlabelled graphs, it is not so straightforward, due to complications of isomorphism (recall that there are typically many adjacency graphs representing a given unlabelled graph). Nonetheless directed graphs typically require more bits to specify their links, and are hence more complex. Exploring directed graphs will be left for future work.
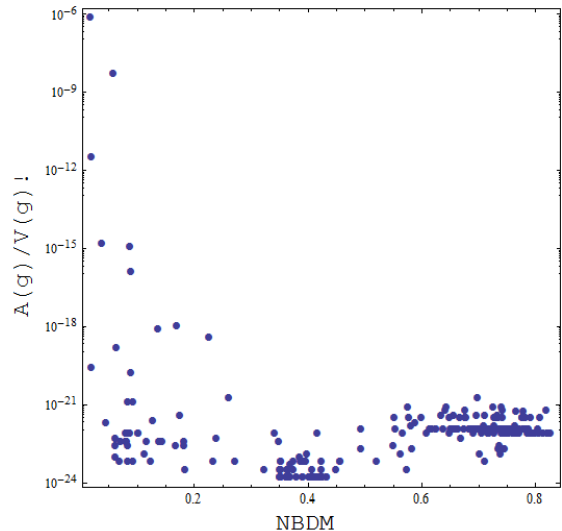
FIG. 8. Real-world networks also display the same correlation between Kolmogorov complexity and automorphism group size $A(g)$. Networks with more symmetries have lower estimated Kolmogorov complexity. Automorphisms count is normalised by network size.

VI. THE ALGORITHMIC RANDOMNESS OF SYNTHETIC COMPLEX NETWORKS

An objective and universal measure of complexity should take into account symmetries as a simplifying factor when it comes to description size. Here we explore how K can characterise topological properties of complex networks.

The study of complex networks is currently an active area of research [2]. The field has been driven largely

by observations that many real-world networks (e.g. internet links or metabolic networks) have properties very different from both regular and random graphs; the latter having been extensively studied in foundational work by Paul Erdös and Alfréd Rényi. Specifically, two topological properties of many complex networks that have been a focus of interest are (a) a scale-free (or power law) distribution in node degree distributions, and (b) the "small-world" property where graphs have high clustering and the average graph distance D grows no faster than the log of the number of nodes: $D \sim \log(V(g))$.

Observations of these properties have motivated the development of many models to explain these features. Of these, The Barabási-Albert model [41] reproduces complex network features using a preferential attachment mechanism, and the Watts-Strogatz model [42] also provides a mechanism for constructing small-world networks with a rewiring algorithm (which involves starting from a regular network and randomly rewiring); see for example Fig. 9.

A. Network connectedness and complexity

We have theoretically substantiated and experimentally demonstrated how network size, both in terms of the number of nodes and the density of edges for a fixed number of nodes, can impact Kolmogorov complexity values (above). However, Fig. 10 demonstrates that node and edge count do not exclusively dominate K, as the graphs in the plot all have exactly the same graph size and edge density. The plot considers Watts–Strogatz graphs of size $V(g) = 1000$ with rewiring probability ranging from $p = 0$ (a regular graph) to $p = 1$ (a random graph). The plot shows that the graph complexity increases with p, thus illustrating a change not subject to graph size or edge density, which are both the same for all cases. Rather, the increasing complexity must be due to other topological properties such as connectedness, link distribution and graph diameter.
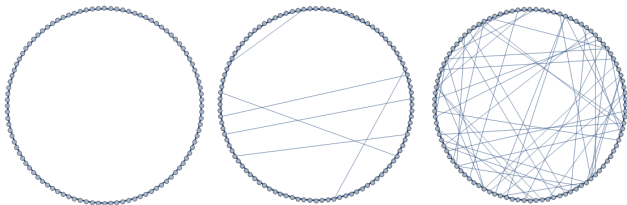

FIG. 9. Example of a Watts–Strogatz rewiring algorithm for $n = 100$-vertex graphs and rewiring probability $p = 0$, .01 and 0.1 starting from a $2n$-regular graph. The larger p the closer to a random graph (rewiring $p = 1$).
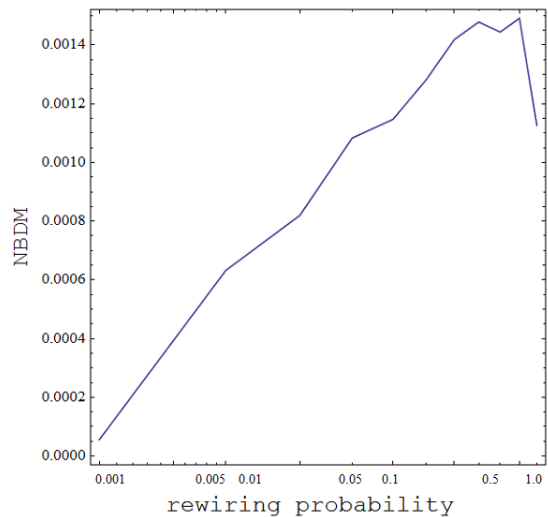

FIG. 10. Kolmogorov complexity of the Watts-Strogatz model as a function of the rewiring probability on a 1000-node network starting from a regular ring lattice. Both the number of nodes and the number of links are kept constant, while p varies; Kolmogorov complexity increases with p.

B. Topological characterization of artificial complex networks

Random networks and Barabási-Albert networks not only have exactly the same vertex count $V(g)$ in the experiment the results of which are summarised in Fig. 12 and 13, but also the same number of edges on average. The K_m difference can only therefore be attributed to other topological properties related to each network model. Observing that K can be affected by topological features such as clustering coefficient (as shown in 11 for Watts-Strogatz networks), we proceeded to examine other network models. As shown in Figs. 12 and 13, K_m approximated by the BDM assigns low Kolmogorov complexity to regular graphs and Watts-Strogatz networks and higher complexity to Barabási-Albert networks, with random networks displaying the greatest Kolmogorov complexity as expected. Indeed, that random graphs are the most algorithmically complex is clear from a theoretical point of view: nearly all long binary strings are algorithmically random, and so nearly all random unlabelled graphs are algorithmically random [19].

Barabási-Albert networks are often referred to as scale free, because the node distribution follows a power law. However for small graphs such as those we analyse, it is questionable whether the global scaling is meaningful. An advantage of the method is, however, that it still differentiates between different network models despite their small size. The theory of algorithmic information formally characterizes any object in terms of the properties specified in its description from which the object can be fully recovered. It is therefore vis-à-vis small objects that the theory presents its greatest challenges,
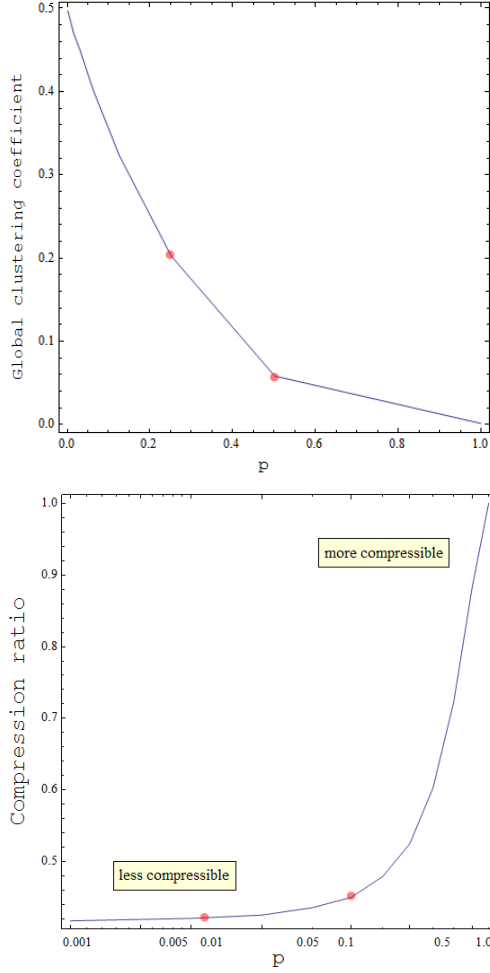
FIG. 11. The Watts-Strogatz model starts from a ring lattice, hence highly compressible, then with rewiring probability p (x-axis) the global clustering coefficient drops fast (Top) in this 15×10^3-node W-S network while approaching a random graph ($p = 1$) slowing down at points $p \sim 0.2$ and 0.5 (red dots in plots) where compression ratios (Bottom) also display slight slope variations.

given that the invariance theorem (see [19]) does not tell us anything about the rate of convergence in values.

We also considered regular graphs for this comparison including Haars, circulants, noncayley transitives, snarks, cubics, books, lattices and suns among other types of regular networks. The average NBDM complexity value of the regular graph group remained very low (and the distribution mostly uniform with a slight peak at around 0.5). The group of complex networks peak at different K_m values, with Watts-Strogatz networks ranking low for $p = 0.01$ when the small world effect is produced, and then moves towards high Kolmogorov complexity when rewiring probability p increases. Barabási-Albert networks peak at NBDM value equal to 0.75 and random graphs (Erdös-Rényi) ranked the highest at an estimated Kolmogorov complexity value close to 0.9.
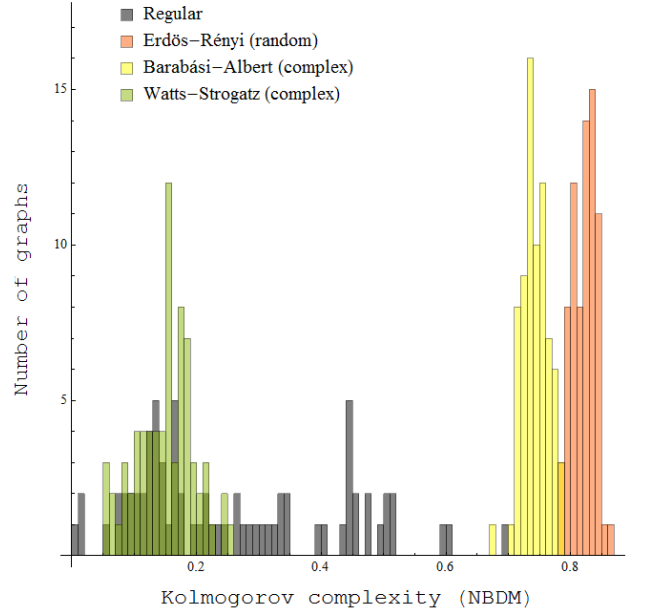
FIG. 12. Distribution of 292 regular, Watts-Strogatz, Barabási-Albert and Erdös-Rényi networks with $V(g) = 30$ (73 networks each) with W-S rewiring probability $p = 0.05$. The number 73 comes from the number of regular graphs of size $V(g) = 30$ in the used repository (*Mathematica*'s Graph-Data[]).
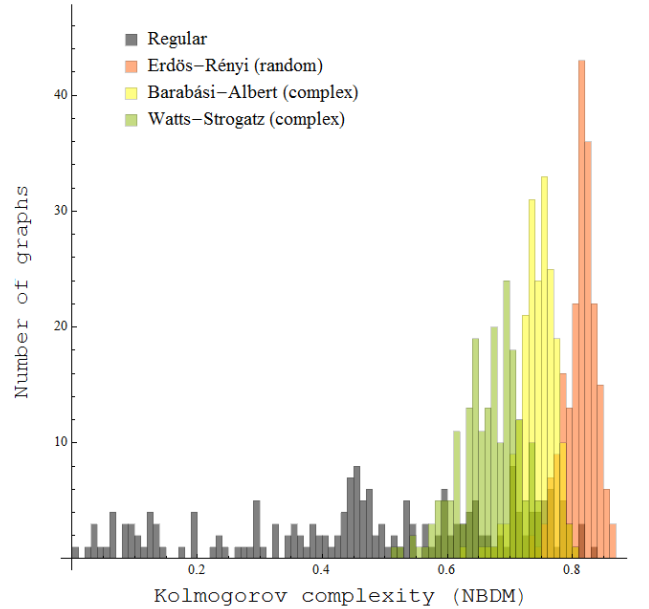
FIG. 13. Distribution of 792 regular, Watts-Strogatz, Barabási-Albert and Erdös-Rényi networks with $V(g) = 20$ (198 networks each) with W-S rewiring probability from $p = 0.05$ in Fig. 12 to $p = 0.5$ as an experiment introducing randomness to witness the shift of the W-S networks towards higher complexity values of random graphs. For combinatorial reasons, there are more regular networks of size 30 than 20, hence this time the number 198 comes from the number of regular graphs found in the (*Mathematica*'s GraphData[]).

VII.   CONCLUDING REMARKS

Our investigation connected disparate seminal subjects in graph theory and complexity science. We have shown computationally that Kolmogorov complexity approximations capture important group-theoretic and topological properties of graphs and networks, properties related to symmetry, density and connectedness, to mention a few and the most important. For example, we found that graphs with a large number of non-trivial automorphism groups tend to have smaller Kolmogorov complexity values. This was verified both for small artificial graphs and larger (but still rather small) empirical networks ranging from biological to social networks.

We have also shown that graphs with differing algorithmic randomness distinguish models of networks, two of which are complex networks with different edge generation mechanisms. A number of connections between graph and network properties to complexity that other papers have claimed or have tried to connect before with hand-tailored measures were obtained naturally with Kolmogorov complexity.

In future work, it may be interesting to take one class of networks of a fixed size and analyse what aspects of the topology change K, and what physical interpretations and implications may be associated to low and high complexities. Additionally, due to the small graphs we have analysed, it would be interesting to explore how approximations to K behave for increasing network size for each model, and also with larger samples of graphs. Finally, extending these results to directed graphs would be another possible direction to explore. The findings suggest that analysing natural complex systems via Kolmogorov complexity may open new interesting avenues of research.

ACKNOWLEDGEMENTS

[1] M. Newman, Networks: An introduction, Oxford University Press, Inc., 2010.

[2] M. Newman, A.-L. Barabasi, D. J. Watts, The structure and dynamics of networks, Princeton University Press, 2011.

[3] D. Bonchev, G. A. Buck, Quantitative measures of network complexity, in: Complexity in Chemistry, Biology, and Ecology, Vol. 7(2), Danail Bonchev and Dennis Rouvray, 2005, pp. 191–235.

[4] J. Kim, T. Wilhelm, What is a complex graph?, Physica A: Statistical Mechanics and its Applications 387 (11) (2008) 2637–2652.

[5] A. Mowshowitz, M. Dehmer, Entropy and the complexity of graphs revisited, Entropy 14 (3) (2012) 559–570.

[6] C. Adami, J. Qian, M. Rupp, A. Hintze, Information content of colored motifs in complex networks, Artificial Life 17 (4) (2011) 375–390.

[7] A. Mowshowitz, V. Mitsou, Entropy, orbits, and spectra of graphs, Analysis of Complex Networks.

[8] Y. Xiao, M. Xiong, W. Wang, H. Wang, Emergence of symmetry in complex networks, Physical Review E 77 (6) (2008) 066108.

[9] Y. Xiao, B. D. MacArthur, H. Wang, M. Xiong, W. Wang, Network quotients: Structural skeletons of complex systems, Physical Review E 78 (4) (2008) 046102.

[10] H. Buhrman, M. Li, J. Tromp, P. Vitányi, Kolmogorov random graphs and the incompressibility method, SIAM Journal on Computing 29 (2) (1999) 590–599.

[11] H. Zenil, F. Soler-Toscano, J.-P. Delahaye, N. Gauvrit, Two-dimensional Kolmogorov complexity and validation of the coding theorem method by compressibility, arXiv 1212:6745 [cs.CC].

[12] J. Hopcroft, R. Tarjan, Efficient planarity testing, J. ACM 21 (4) (1974) 549–568.

[13] R. Read, D. Corneil, The graph isomorphism disease, Journal of Graph Theory 1 (1) (1977) 339–363.

[14] K. Baskerville, M. Paczuski, Subgraph ensembles and motif discovery using an alternative heuristic for graph isomorphism., Phys Rev E Stat Nonlin Soft Matter Phys 74 (5 Pt 1) (2006) 051903.

[15] K. Baskerville, P. Grassberger, M. Paczuski, Graph animals, subgraph sampling, and motif search in large networks, Physical Review E 76 (3) (2007) 36107.

[16] M. Girvan, M. E. J. Newman, Community structure in social and biological networks, Proceedings of the National Academy of Sciences 99 (12) (2002) 7821–7826. doi:10.1073/pnas.122653799.

[17] A. N. Kolmogorov, Three approaches to the quantitative definition of information, Problems of Information and Transmission 1 (1) (1965) 1–7.

[18] G. J. Chaitin, On the length of programs for computing finite binary sequences: Statistical considerations, Journal of the ACM 16 (1) (1969) 145–159.

[19] M. Li, P. Vitányi, An Introduction to Kolmogorov Complexity and Its Applications, Springer, Heidelberg, 2008.

[20] The group of valid programs forms a prefix-free set (no element is a prefix of any other, a property necessary to keep $0 < m(s) < 1$.) For details see [24].

[21] R. J. Solomonoff, A formal theory of inductive inference: Parts 1 and 2, Information and Control 7 (1-22) (1964) 224–254.

[22] L. Levin, Laws of information conservation (non-growth) and aspects of the foundation of probability theory, Problems in Form. Transmission 10 (1974) 206–210.

[23] T. M. Cover, J. A. Thomas, Information Theory, 2nd Edition, J. Wiley and Sons.

[24] C. S. Calude, Information and Randomness, Springer, 2002.

[25] J.-P. Delahaye, H. Zenil, On the Kolmogorov-Chaitin complexity for short sequences, in: C. Calude (Ed.), Randomness and Complexity: From Leibniz to Chaitin, World Scientific, 2007.

[26] J.-P. Delahaye, H. Zenil, Numerical evaluation of the complexity of short strings: A glance into the innermost structure of algorithmic randomness, Applied Math. and Comp 219 (2012) 63–77.

[27] H. Zenil, Une approche expérimentale à la théorie algorithmique de la complexité, dissertation in fulfilment of the degree of doctor in computer science, Ph.D. thesis, Université de Lille 1 (June 2011).

[28] J.-P. Delahaye, H. Zenil, Numerical evaluation of the complexity of short strings: A glance into the innermost structure of algorithmic randomness, Applied Mathematics and Computation 219 (2012) 63–77.

[29] H. Zenil, J.-P. Delahaye, On the algorithmic nature of the world, in: Information and Computation, G. Dodig-Crnkovic and M. Burgin (eds), World Scientific Publishing CompanyWorld Scientific Publishing Company, 2010.

[30] F. Soler-Toscano, H. Zenil, J.-P. Delahaye, N. Gauvrit, Calculating Kolmogorov complexity from the frequency output distributions of small Turing machines, arXiv 1211:1302 [cs.IT].

[31] F. Soler-Toscano, H. Zenil, J.-P. Delahaye, N. Gauvrit, Correspondence and independence of numerical evaluations of algorithmic information measures, Computability (in press).

[32] H. Zenil, E. Villarreal-Zapata, Asymptotic behaviour and ratios of complexity in cellular automata rule spaces, Journal of Bifurcation and Chaos (2013) (in press).

[33] T. Radò, On non-computable functions, Bell System Technical Journal 41 (3) (1962) 877–884.

[34] A. H. Brady, The determination of the value of Rado's noncomputable function $Sigma(k)$ for four-state turing machines, Mathematics of Computation 40, 162 (1983) 647–665.

[35] C. Langton, Studying artificial life with cellular automata, Physica D: Nonlinear Phenomena 22 (1-3) (1986) 120–149.

[36] M. Gell-Mann, The Quark and the Jaguar: Adventures in the Simple and the Complex, Abacus; New edition edition, Paris, 1995.

[37] M. A. Dehmer M, A history of graph entropy measures, Information Sciences 181 (2011) 5778.

[38] M. Dehmer, L. Sivakumar, Recent developments in quantitative graph theory: Information inequalities for networks, PLoS One 7(2) (2012) 5778.

[39] R. K. Standish, Complexity of networks, in: Recent Advances in Artificial Life, Abbass et al. (eds), World Scientific Publishing CompanyWorld Scientific Publishing Company, 2005, p. 253.

[40] H. Katebi, K. A. Sakallah, I. L. Markov, Conflict anticipation in the search for graph automorphisms, in: Int'l Conf. on Logic for Programming, LPAR, Merida, Venezuela., 2012.

[41] R. Albert, A.-L. Barabási, Statistical mechanics of complex networks, Reviews of modern physics 74 (1) (2002) 47.

[42] D. J. Watts, S. H. Strogatz, Collective dynamics of 'small-world' networks, Nature 393 (6684) (1998) 409–10.

[43] H. Jeong, B. Tombor, R. Albert, Z. Oltvai, A.-L. Barabási, The large-scale organization of metabolic networks, Nature 407 (2000) 651.

[44] D. Johnson, M. T. (eds), Cliques, Coloring, and Satisfiability: Second DIMACS Implementation Challenge, Vol. 26, AMS, Providence, RI, USA, 1996.

APPENDIX

This Appendix contains some additional figures to the main text.

FIG. 14. Plots of number of graph automorphisms normalised by maximum number of edges of g, $A(g)/V(g)!$ (y-axis) versus (normalised) Kolmogorov complexity (x-axis) estimated by NBDM for connected regular graphs found in *Mathematica* (Graph-Data[]) with size $V(g) = 20$ to 36 nodes (only vertex sizes for which at least 20 graphs were found in the dataset were plotted). The decay can be seen, though the relationship is noisy.
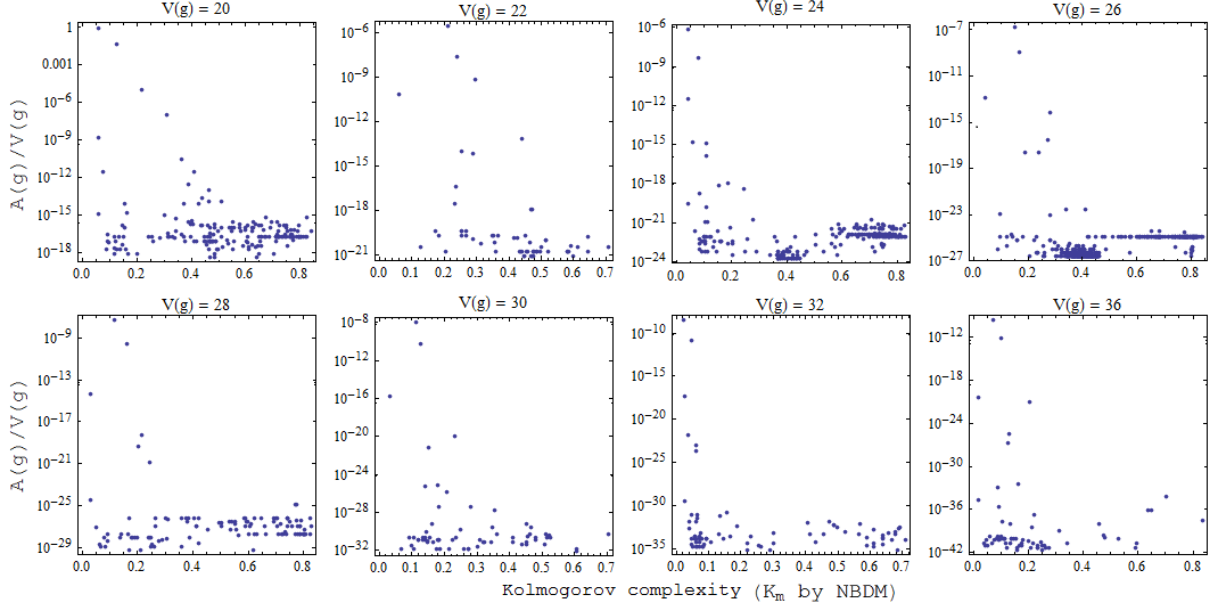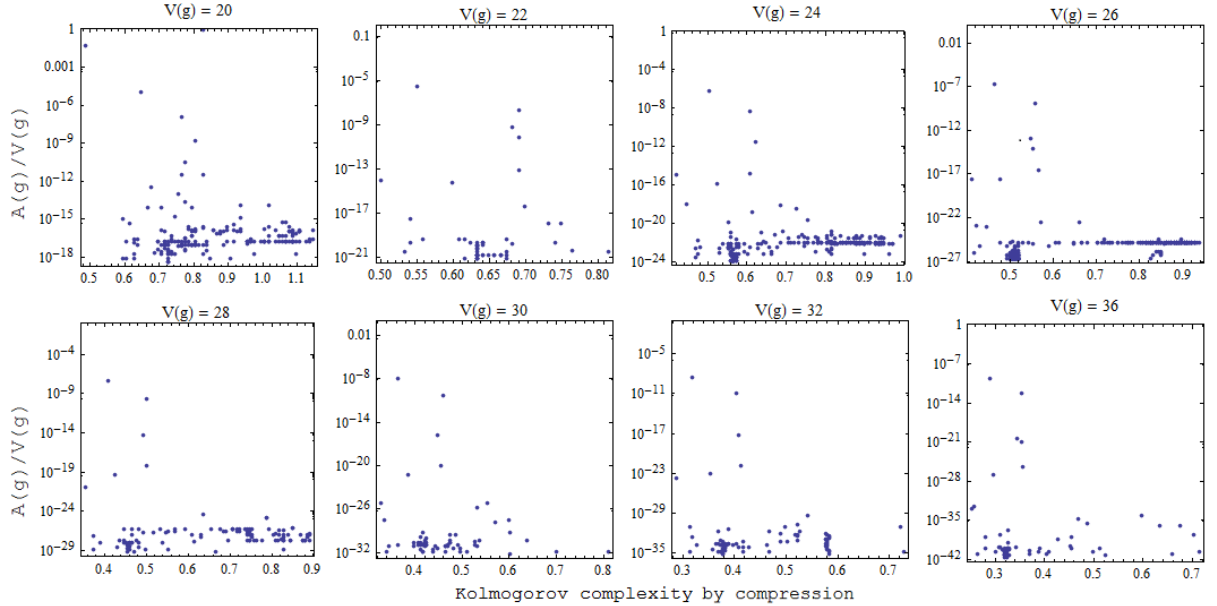

FIG. 15. Plots of number of graph automorphisms normalised by maximum number of edges of g, $A(g)/V(g)!$ (y-axis) versus Kolmogorov complexity (x-axis) estimated by lossless compressed length (Deflate) of connected regular graphs in *Mathematica*'s GraphData[] with size $V(g) = 20$ to 36 nodes (only vertex sizes for which at least 20 graphs were found in the dataset were plotted).

Network description (g)	$V(g)$	Normalised $K_m(g)$ (BDM)	$A(g)/V(g)$
Metabolic Network of Actinobacillus Actinomycetemcomitans	993	0.00336	4.42×10^{74}
Metabolic Network Neisseria Meningitidis	981	0.00344	2.86×10^{76}
Perl Module Authors Network	840	0.00350	4.63×10^{470}
Metabolic Network Campylobacter Jejuni	946	0.00370	6.97×10^{74}
Metabolic Network Emericella Nidulans	916	0.00378	3.43×10^{68}
Pyrococcus Horikoshii Network	953	0.00382	4.22×10^{70}
Pyrococcus Furiosus Network	931	0.00384	3.37×10^{68}
Metabolic Network Thermotoga Maritima	830	0.00477	2.05×10^{64}
Mycoplasma Genitalium Network	878	0.00480	6.75×10^{92}
Treponema Pallidum Network	899	0.00499	2.71×10^{84}
Chlamydia Trachomatis Network	822	0.00511	1.73×10^{75}
Metabolic Network Pyrococcus Furiosus	751	0.00511	2.86×10^{50}
Rickettsia Prowazekii Network	817	0.00523	1.39×10^{76}
Arabidopsis Thaliana Network	768	0.00535	1.93×10^{60}
Oryza Sativa Network	744	0.00569	3.45×10^{57}
Chlamydia Pneumoniae Network	744	0.00635	2.00×10^{70}
Metabolic Network Oryza Sativa	665	0.00640	9.49×10^{47}
Metabolic Network Rickettsia Prowazekii	456	0.01080	1.10×10^{34}
Metabolic Network Mycoplasma Pneumoniae	411	0.01280	1.85×10^{28}
Metabolic Network Borrelia Burgdorferi	409	0.01460	2.10×10^{36}

TABLE I. Random sample of 20 real-world networks [43, 44] from the 88 included in the study (and plotted in Fig. 8), sorted from smallest to largest estimated Kolmogorov complexity values (NBDM). While the (negative) correlation between K_m and $V(g)$ is almost perfect (Pearson coefficient -0.95) the (negative) correlation between K_m and $A(g)$ is significant (Pearson coefficient -0.178) after normalisation by $V(g)$ for the 88 elements. The full descriptions and sources of the networks are available as supplemental material at http://www.complexitycalculator.com/graphcomplexitysuplementalmaterialv1.zip.

The correlation of K_m and $V(g)$ is explained by the theory. K (not K_m but the true uncomputable value K) is strongly positively correlated to $V(g)$ because larger networks can always potentially reach higher complexity values compared to small networks.

Row 1:
- V(w) = 30, E(w) = 60, D(w) = 6, C(w) = 0.335106, $K_m \log(w) = 797.472$
- V(w) = 30, E(w) = 60, D(w) = 5, C(w) = 0.315789, $K_m \log(w) = 813.535$
- V(w) = 30, E(w) = 60, D(w) = 6, C(w) = 0.267016, $K_m \log(w) = 844.603$
- V(b) = 30, E(b) = 57, D(b) = 4, C(b) = 0.170103, $K_m \log(b) = 859.963$
- V(w) = 30, E(w) = 60, D(w) = 5, C(w) = 0.260204, $K_m \log(w) = 862.968$
- V(w) = 30, E(w) = 60, D(w) = 5, C(w) = 0.182741, $K_m \log(w) = 870.048$

Row 2:
- V(w) = 30, E(w) = 60, D(w) = 5, C(w) = 0.285714, $K_m \log(w) = 880.706$
- V(w) = 30, E(w) = 60, D(w) = 5, C(w) = 0.237624, $K_m \log(w) = 885.761$
- V(w) = 30, E(w) = 60, D(w) = 5, C(w) = 0.188482, $K_m \log(w) = 889.341$
- V(w) = 30, E(w) = 60, D(w) = 5, C(w) = 0.226131, $K_m \log(w) = 902.811$
- V(w) = 30, E(w) = 60, D(w) = 5, C(w) = 0.246154, $K_m \log(w) = 905.58$
- V(w) = 30, E(w) = 60, D(w) = 5, C(w) = 0.178218, $K_m \log(w) = 910.027$

Row 3:
- V(w) = 30, E(w) = 60, D(w) = 5, C(w) = 0.186528, $K_m \log(w) = 918.057$
- V(w) = 30, E(w) = 60, D(w) = 4, C(w) = 0.171875, $K_m \log(w) = 920.222$
- V(w) = 30, E(w) = 60, D(w) = 6, C(w) = 0.168367, $K_m \log(w) = 922.214$
- V(w) = 30, E(w) = 60, D(w) = 5, C(w) = 0.161765, $K_m \log(w) = 935.726$
- V(w) = 30, E(w) = 60, D(w) = 6, C(w) = 0.157895, $K_m \log(w) = 946.841$
- V(w) = 30, E(w) = 60, D(w) = 5, C(w) = 0.191176, $K_m \log(w) = 949.376$

Row 4:
- V(w) = 30, E(w) = 60, D(w) = 5, C(w) = 0.188482, $K_m \log(w) = 967.149$
- V(w) = 30, E(w) = 60, D(w) = 5, C(w) = 0.24878, $K_m \log(w) = 975.019$
- V(w) = 30, E(w) = 60, D(w) = 6, C(w) = 0.203125, $K_m \log(w) = 979.723$
- V(w) = 30, E(w) = 60, D(w) = 5, C(w) = 0.158654, $K_m \log(w) = 1003.16$
- V(b) = 30, E(b) = 57, D(b) = 4, C(b) = 0.166052, $K_m \log(b) = 1007.47$
- V(b) = 30, E(b) = 57, D(b) = 4, C(b) = 0.185455, $K_m \log(b) = 1019.04$

Row 5:
- V(b) = 30, E(b) = 57, D(b) = 4, C(b) = 0.148855, $K_m \log(b) = 1031.9$
- V(w) = 30, E(w) = 60, D(w) = 5, C(w) = 0.173077, $K_m \log(w) = 1042.12$
- V(w) = 30, E(w) = 60, D(w) = 6, C(w) = 0.231884, $K_m \log(w) = 1043.9$
- V(w) = 30, E(w) = 60, D(w) = 5, C(w) = 0.19598, $K_m \log(w) = 1050.45$
- V(b) = 30, E(b) = 57, D(b) = 5, C(b) = 0.180723, $K_m \log(b) = 1052.14$
- V(w) = 30, E(w) = 60, D(w) = 4, C(w) = 0.114286, $K_m \log(w) = 1091.88$

Row 6:
- V(b) = 30, E(b) = 57, D(b) = 5, C(b) = 0.156522, $K_m \log(b) = 1107.2$
- V(b) = 30, E(b) = 57, D(b) = 4, C(b) = 0.165441, $K_m \log(b) = 1124.94$
- V(r) = 30, E(r) = 78, D(r) = 4, C(r) = 0.0980926, $K_m \log(r) = 1185.18$
- V(r) = 30, E(r) = 76, D(r) = 4, C(r) = 0.167553, $K_m \log(r) = 1189.03$
- V(r) = 30, E(r) = 74, D(r) = 4, C(r) = 0.117117, $K_m \log(r) = 1199.37$
- V(r) = 30, E(r) = 68, D(r) = 5, C(r) = 0.16129, $K_m \log(r) = 1212.31$

FIG. 16. Random (Erdös–Rényi) graphs (denoted by r) versus complex networks (Watts-Strogatz and Barabási-Albert) (denoted by w and b) sorted by Kolmogorov complexity (smallest to largest $K_m \log$) as approximated by the BDM.