nature machine intelligence

SUPPLEMENTARY INFORMATION

https://doi.org/10.1038/s42256-018-0005-0

In the format provided by the authors and unedited.

Causal deconvolution by algorithmic generative models

Hector Zenil[1,2,3,4,5,6*], Narsis A. Kiani[1,3,4,6*], Allan A. Zea[1,4,7] and Jesper Tegnér[3,5,6*]

[1]Algorithmic Dynamics Lab, Unit of Computational Medicine, Center for Molecular Medicine, Karolinska Institutet, Stockholm, Sweden. [2]Oxford Immune Algorithmics, Oxford University Innovation, Oxford, UK. [3]Unit of Computational Medicine, Center for Molecular Medicine, Department of Medicine, Karolinska Institutet, Stockholm, Sweden. [4]Algorithmic Nature Group, LABORES for the Natural and Digital Sciences, Paris, France. [5]Living Systems Laboratory, Biological and Environmental Sciences and Engineering Division and Computer, Electrical and Mathematical Sciences and Engineering Division, King Abdullah University of Science and Technology, Jeddah, Saudi Arabia. [6]Science for Life Laboratory, Karolinska Institutet, Stockholm, Sweden. [7]Escuela de Matemática, Facultad de Ciencias, UCV, Caracas, Venezuela. *e-mail: hector.zenil@ki.se; narsis.kiani@ki.se; jesper.tegner@kaust.edu.sa

Supplementary Information

Supplementary Methods

Cellular Automata

Definition 0.1. A *cellular automaton* (or CA) is a tuple $\langle S, (\mathbb{L}, +), T, f\rangle$ with a set S of states, a lattice $\mathbb{L}$ with a binary operation $+$, a neighbourhood template T, and a local rule f.

The *set of states* S is a finite set with elements s taken from a finite alphabet $\sum$ with at least 2 elements.

Definition 0.2. The *neighbourhood template* $T = \langle \eta_1, \ldots, \eta_m\rangle$ is a sequence of $\mathbb{L}$. In particular, the neighbourhood of cell i is given by adding the cell i to each element of the template T: $T = \langle i + \eta_1, \ldots, i + \eta_m\rangle$. Each cell i of the CA is in a particular state $c[i] \in S$. A *configuration* of the CA is a function $c : \mathbb{L} \to S$. The *set of all possible configurations* of the CA is defined as $S_{\mathbb{L}}$.

As a discrete dynamical system, the *evolution of the CA* occurs in discrete time steps $t = 0, 1, 2, \ldots, n$. The transition from a configuration c_t at time t to the configuration $c_{(t+1)}$ at time $t + 1$ is induced by applying the local rule f. The local rule is to be taken as a function $f : S^{|T|} \to S$ which maps the states of the neighbourhood cells of time step t in the neighbourhood template T to cell states of the configuration at time step $t + 1$:

$$c_{t+1}[i] = f\left(c_t[i + \eta_1], \ldots, c_t[i + \eta_m]\right) \tag{1}$$

The general transition from configuration to configuration is called the *global map* and is defined as: $F : S^{\mathbb{L}} \to S^{\mathbb{L}}$. The code in Wolfram Language is available in [8]

Enumeration of ECA rules

In the case of 1-dimensional CA, it is common to introduce the *radius* of the neighbourhood template which can be written as $\langle -r, -r+1, \ldots, r-1, r\rangle$ and has length $2r + 1$ cells. With a given radius r the local rule is a function $f : \mathbb{Z}_{|S|}^{|S|^{(2r+1)}} \to \mathbb{Z}_{|S|}$ with $\mathbb{Z}_{|S|}^{|S|^{(2r+1)}}$ rules. *Elementary Cellular Automata* or ECA, have a radius $r = 1$ (closest neighbours), having the neighbourhood template $\langle -1, 0, 1\rangle$, meaning that the neighbourhood comprises a central cell. From this it follows that the rule space for ECA contains $2^{2^3} = 256$ rules.

It is common to follow a lexicographic ordering scheme, as introduced by Wolfram [1], in the enumeration of CA and ECA. According to this scheme, the 256 ECA rules can be encoded by only 8-bits, once part of the rule is fixed for all of them.

Randomly Interacting Cellular Automata

Our interactive Cellular Automaton model, as introduced in [2], is of such a nature that only one of them survives the contact between the CA colours, or else they both disappear. This means that there are only 3 possible solutions to what happens when cells of both CA come into contact in the same neighbourhood, and that there is in reality a single controlling CA with 3 colours that governs the interaction of the other 2 whose local rules dictate that: grey survives, black survives, or only a white cell remains. However, grey does not need to be displayed because it is only auxiliary and determines what happens when cells of the different CA come in contact in the same neighbourhood.

In particular we have it that $c_{t+1}(x_n)$ should be either white or black in case none of $c_t(x_{n-1})$, $c_t(x_n)$ and $c_t(x_{n+1})$ is grey. Likewise, $c_{t+1}(x_n)$ should be either white or grey in case none of $c_t(x_{n-1})$, $c_t(x_n)$ and $c_t(x_{n+1})$ is black. For example, let $N = \langle c_t(x_{j-1}), c_t(x_j), c_t(x_{j+1})\rangle$ and $M = \langle c_{t'}(x_{i-1}), c_{t'}(x_i), c_{t'}(x_{i+1})\rangle$ be 2 different mixed neighbourhoods. Then there is no correlation between the random values of $c_{t+1}(x_j)$ and of $c_{t'+1}(x_i)$. Note that we impose this independence both in case $N = M$ (so that the difference is only reflected in either the location (that is, $i \neq j$) or in the time ($t \neq t'$)) and in case $N \neq M$. In particular, the mixed neighbourhood $\langle 2, 2, 1\rangle$ may sometimes yield a 0, sometimes a 1, and at yet other times a 2.

In [3] and [4] these interactions are investigated in terms of evolution and complexity. We also provide 2 more examples of interacting CA and a comparison with other methods and measures, in particular the Partial Information Decomposition [17] using both classical Mutual Information [17] and the Normalised Compression Distance [22]. One striking difference from these other methods is that ours does not require any associated probability distributions or arbitrary parameter choice, including the terminating criterion explained in Subsection 3.2.1 (main text).

2-System Interactions

The qualitative behaviour of each program can heuristically be identified by what is known as its Wolfram class, which in turn has been formalised using tools and methods from algorithmic complexity in [5, 6]. Informally, Wolfram class 1 represents evolutions of programs that converge to a simple fixed configuration, exemplars of Wolfram class 2 converge to repetitive simple behaviour, those of Wolfram class 3 produce unbounded, apparently statistically random behaviour, and exemplars of Wolfram class 4 reproduce apparently open-ended persistent structures. None of what has been introduced here depends on this behavioural characterisation based on different heuristics, and it is thus in no way fundamental to the results reported.

For our purposes, and to avoid any bias due to white background, we start systems (ECA) from initial conditions spanning the width of the evolution producing a

squared or rectangular space-time diagram Figure 2c rather than the typical pyramidal evolution that starts from the simplest initial condition (a black cell), such as the examples in Figs. 2a,b. Figure 3 illustrates the algorithm's success at decomposing different 2-system interactions/compositions even with similar qualitative behaviour and statistics, thus making it difficult for other statistical methods to perform the same task.

The following source code implements an algorithm determining the way in which local CA are dictated and governs their intersection. It can be implemented with this Wolfram Language (Mathematica) code:

```
R[x_]:=Thread[Rule[{{-1,1,0},{-1,0,1},{-1,1,1},{1,-1,1},{1,-1,0},
{1,1,-1},{1,0,-1}, {0,1,-1},{0,-1,1}, {1,-1,-1},{-1,1,-1},
{-1,-1,1}},Flatten[Take[Tuples[{-1,0,1},12],{x,x}]]]];
Code[n_]:=BitXor[n,BitShiftRight[n]];
RuleCode[n_]:=IntegerDigits[Code[n],2]
```

where *RuleCode*[] retrieves a rule governing the interaction between any 2 CA.

Alternative code can be found at the Wolfram Demonstrations website at `http://demonstrations.wolfram.com/CompetingCellularAutomata/` [8].

Deconvolution algorithms

Main algorithm pseudocode

Let us denote the number of components of G by $k(G)$.

Algorithm 1 Causal deconvolution algorithm

function DECONVOLVE(G, N), $1 \leq k(G) \leq N \leq |V(G)|$
  while $k(G) < N$ **do**
    $informationLoss \leftarrow \varnothing$
    // for each edge e
    for $e \in E(G)$ **do**
      if $I(G, e) > 0$ **then**
        // store information contribution into *informationLoss*
        $informationLoss \leftarrow informationLoss \cup \{I(G, e)\}$
    // calculate the minimal information loss across all edges
    $minLoss \leftarrow \min(informationLoss)$
    // remove all candidate edges from G
    $G \leftarrow G \backslash \{e \in E(G) : I(G, e) = minLoss\}$
  return G

The algorithm introduced is independent of the method used to approximate algorithmic complexity, such as BDM. BDM assigns an index associated with the

size of the most likely generating mechanism producing the data according to Algorithmic Probability [15]. BDM is capable of capturing features in data beyond statistical properties [39, 50], and thus represents an improvement over classical information theory. Because finding the program that reproduces a large object is computationally very expensive—even to approximate—BDM finds short candidate programs using another method [27, 28] that finds and reproduces fragments of the original object and then puts them together as a candidate algorithmic model of the whole object [39, 43]. These short computer programs are effectively candidate models explaining each fragment, with the long finite sequence of short models being itself a generating mechanistic model.

Time Complexity The algorithms for network deconvolution introduced in this section run in polynomial time after the precalculation of estimations to algorithmic probability [27, 28]. Let M denote the number of edges of the graph G. The brute force algorithm for this problem searches the edge such that its removal minimises the loss of information and deletes it, repeating this process for all edges of G until N subcomponents are reached, which has a worst-case time complexity of $O(M^2)$. Algorithm 1 is different from the brute force approach in that edges with equal minimal contributions to the loss of information are not deleted sequentially but all at once, but its time complexity is also of $O(M^2)$. The outer loop that verifies whether the number of desired subcomponents is reached is removed in Algorithm 2, allowing us to find the optimal terminating step of the deconvolution in time $O(M)$.

Algorithm with automatic terminating criterion

The term ε is related to the number of components N from Algorithm 1, or rather substitutes for N. ε is an auxiliary cutoff value that determines when to stop the algorithm without making an arbitrary choice of number of subcomponents N. ε is 0 for the theoretical cutoff value $\log(2)$ that determines the effect of perturbations performed on the network. If removing an edge has an effect greater than $\log(2)$, then such an edge does not belong to the same underlying algorithm explaining the rest of G, given that the program size of a deterministic object generated by the same computer program does not change by more than $\log(2)$, and thus nor does its algorithmic complexity. In contrast, if the perturbation by edge removal has a loss of at most $\log(2)$ bits, then it means that it is likely to be reconstructed by the original computer program, because $\log(2)$ is the growth in the description of a computer program (or a deterministic system) that accounts only for running time. In other words, if the perturbation is above $\log(2)$, it means that such an edge may have disconnected 2 or more causally independent components with different computer programs likely able to explain each different subcomponent with fewer bits than when keeping those components together using such an edge.

Algorithm 2 Causal deconvolution with terminating criterion

function DECONVOLVE(G, ε),
  $informationLoss \leftarrow \varnothing$
  // for each edge e
  for $e \in E(G)$ **do**
    if $I(G, e) > 0$ **then**
      // store information contribution into *informationLoss*
      $informationLoss \leftarrow informationLoss \cup \{I(G, e)\}$
  for $loss \in informationLoss$ **do**
    $difference \leftarrow 0$
    if $|informationLoss| > 1$ **then**
      // copy maximum loss to *maxLoss*
      $maxLoss \leftarrow \max(informationLoss)$
      $informationLoss \leftarrow informationLoss \backslash \{maxLoss\}$
      // calculate difference between the old and new maxima
      $difference \leftarrow maxLoss - \max(informationLoss)$
      // if *difference* significantly departs from log(2)
      if $|difference - \log(2)| > \varepsilon$ **then**
        // remove all candidate edges from G
        $G \leftarrow G \backslash \{e \in E(G) : I(G, e) = \max(informationLoss)\}$
  return G

In practice, however, such a strict cutoff value does not occur, so ε accounts for a difference or error not far from log(2). Moreover, ε can be estimated from the sequential information differences calculated from the absolute distances between the differences of consecutive values in the information signature (the list of information values for all edges sorted by maximum contribution) and its deviation from log(2), so no cut is made for an edge with information difference below $\log(2) + \varepsilon$ (see Figure 4).

Other Methods and Measures

Figure 1 shows the results obtained by replacing the algorithmic probability estimations (BDM) in our *causal decomposition* method by classical information theory (Shannon entropy) 1b and one of the most popular lossless compression algorithms (Compress) 1c, based on LZW as an approximation of algorithmic (Kolmogorov-Chaitin) complexity instead of BDM. Because all values collapse into a single value for entropy, the colours displayed are the result of an artificial sorting of the pixels based on their indices, from top to bottom. Compression is a lower-quality approximation of what we reported in Figs. 2c-f, where the reported algorithm based on

the BDM is clearly an improvement.

In Figure 3 and Figure 4, we compared the accuracy and sensitivity of our *causal deconvolution* algorithm introduced here against 2 other methods based on the concept of Partial Information Decomposition [17] (PID) based on (classical) Mutual Information and also lossless compression by means of the Normalised Compression Distance [22] [22].

The original PID [17] method requires us to take all subsets of the *predictor variables* in X, with X set of all predictor variables of a target variable S (e.g. the space-time diagram of interacting CA), and equivalent to our action of pixel perturbation.

Beyond the fact that the PID and NCD methods are based on different first principles, the main problem with both approaches seems to be a problem of sensitivity. In Figure 4, we tested how many values (grey shades) each method was able to produce as an indication of how much each method would fine- or coarse-grain the original 2-system interaction.

It was clear that the original version of PID was intractable, as it was impossible to deal with all the subsets of pixel perturbations of the interacting CA space-time diagram in Figure 4a even for only 100 steps. So we proceeded by applying perturbations to rows, only taking subsets of 6 bits at a time in a sliding non-overlapping window traversing each row left to right.

The original PID algorithm is based on (classical) Mutual Information (thus Shannon entropy). $I(X;S)$ then provides values for redundant and synergistic information between all subsets of X. The results in Figure 4 show the sensitivity of all 3 methods, demonstrating that all can capture features of the interactions, but with different sensitivity. The Normalised Compression Distance (Figure 4c) as introduced in [22], and based on the algorithm Compress, in turn based on LZW, was the least sensitive, retrieving less than 5 different values to recolour the system and thus heavily coarse-graining most features of Figure 4a. Neither this adapted version of PID nor the version with Normalised Compression Distance were able to separate regions as done in 2.

Causal deconvolution (Figure 4d) was the most sensitive, capturing more structure from the original 2-system interaction by virtue of finer grained capabilities. Moreover, as established in the main text, the underlying principles of each method are fundamentally different. Mutual Information, and thus Shannon entropy, can only quantify statistical regularities, while popular lossless compression is heavily related to Shannon entropy [35], despite its wide use as a method for approximating algorithmic complexity. These methods were not able to separate the interacting CA as our method did (Figure 2), due to lack of sensitivity and specificity.

Graph Complexity

The concept of *Algorithmic Probability* (and of Levin's semi-measure, and the universal distribution associated with it) has been introduced as a method for approximating algorithmic complexity based on the frequency of the patterns occurring in the adjacency matrix of a graph [49, 7]. The measure applied to labelled graphs has been proven to be a tight upper bound of the algorithmic complexity of unlabelled graphs and therefore quite invariant to particular adjacency matrix choice.

More precisely, the algorithmic probability [15, 41, 40] of a subgraph $H \subseteq G$ is a measure of algorithmic probability based on the frequency of a random computer program p producing H when run on a 2-dimensional tape universal (prefix-free[1]) Turing machine U also referred to as a *Turmite* [9]. That is,

$$m(G) = \sum_{p:U(p)=H\subseteq G} 1/2^{|p|}. \tag{2}$$

The probability semi-measure $m(G)$ is related to algorithmic complexity $C(G)$ in that $m(G)$ is at least the maximum term in the summation of programs $m(G) \geq 2^{-C(G)}$, given that the shortest program carries the greatest weight in the sum.

The algorithmic complexity (also known as Kolmogorov-Chaitin complexity) [48, 40] is defined as the length of the shortest computer program that reproduces the data from its compressed form when running on a universal Turing machine.

The coding theorem [15, 41] establishes the connection between $m(G)$ and $C(G)$ as $|-\log_2 m(G) - C(G)| < c$, where c is some fixed constant independent of s. The theorem implies that one can estimate the algorithmic complexity of a graph from its frequency of production by running random programs and applying the coding theorem: $C(G) = -\log_2 m(G) + c$. We call this approach the Coding Theorem Method (CTM). The coding theorem establishes that graphs produced with lower frequency by random computer programs have higher algorithmic complexity, and vice versa. Applying the so-called Coding Theorem Method (CTM) [27, 28] and the Block Decomposition Method (BDM) [43, 39], based on estimations of algorithmic probability, involves running a very large number of small computer programs according to a quasi-lexicographic order (from smaller to larger program size) to produce an empirical approximation of the universal distribution m. The BDM of a graph then consists in decomposing its adjacency matrix into subgraphs of sizes for which complexity values estimated by CTM are available, then reconstructing a sequence model that can explain the full G by assembling in sequence the smaller models that produce all the parts of G in combination with rules from classical information theory, as follows:

[1]The group of valid programs forms a prefix-free set (no element is a prefix of any other, a property necessary to keep $0 < m(G) < 1$). Because $m(G) < 1$, $m(s)$ is called a semi-measure, because not all programs halt and thus it never reaches 1.

$$C(G) = \sum_{(r_u, n_u) \in Adj(G)_{d \times d}} \log_2(n_u) + C(r_u) \tag{3}$$

where $Adj(G)_{d \times d}$ represents the set with elements (r_u, n_u), obtained when decomposing the adjacency matrix of G into all subgraphs of size d contained in G. In each (r_u, n_u) pair, r_u is one such submatrix of the adjacency matrix and n_u its multiplicity (number of occurrences). As can be seen from the formula, repeated subgraphs only contribute to the complexity value with the subgraph BDM complexity value once plus a logarithmic term as a function of the number of occurrences. This is because the information content of subgraphs is only sub-additive, as one would expect from the growth of their description lengths. Applications of $m(G)$ and $C(G)$ have been explored in [27, 28, 43], and include applications to graph theory and complex networks [10] and in [43] where the technique was first introduced.

The only parameter used for the application of BDM, as suggested in [39], is the overlap of the decomposition, which is set to the maximum 12 bits for strings and 4 square bits for arrays, given the current best CTM approximations [28] from an empirical distribution based on all Turing machines with up to 5 states, with no string/array overlapping in the decomposition for maximum efficiency (as it runs in linear time), and for which the error (due to boundary conditions) has been shown to be bounded [39].

Graph Generation for Deconvolution Examples The graphs used throughout this paper were generated using the Wolfram Language on the Mathematica platform using the function RandomGraph[] with uniform distribution (UniformGraphDistribution[]) for Erdős-Rényi graphs and a scale-free distribution (BarabasiAlbertGraphDistribution[]) for the scale-free networks constructed by starting from a cycle graph of size 3, with a vertex of k edges added at each step according to the preferential attachment algorithm [46], following a distribution proportional to the vertex degree. All experiments were replicated 20 times and the results were aggregated and averaged.

0.0.1 Robustness and Limitations

Figure 3d represents a test case to evaluate the effect of additive noise by connecting an E-R graph of increasing size and with an increasingly greater number of random edges.

To find how much structure, if any, can be recovered/extracted when adding a random (E-R) graph to different types of structured networks, we conducted a series of numerical experiments shedding light on the limitations of the algorithm introduced here in the face of additive noise. The same results were obtained for

the simpler case of connecting any complete graph of increasing size to any other, such as an E-R or S-F graph.

Figure 5 shows the results from the experiments separating graphs, in this case, a scale-free graph (S-F) from an Erdős-Rényi graph (E-R), the former generated by a Barábasi-Albert preferential attachment algorithm [46] and the latter produced by a pseudo-random generator. Figure 5a quantifies the error and optimal signal-to-noise ratio for optimal deconvolution, testing the algorithm under additive noise both for fixed and growing size subcomponents. Figure 5b shows links coloured in red as identified by the algorithm having the highest algorithmic information value when their removal sends the original composed system towards lower algorithmic information content, thereby telling apart the 2 subcomponents. Negative links are mostly on the side of the E-R graph. In other words, the S-F can be extracted with the greatest precision and a lower rate of false positives from the mix, which is to be expected given the random nature of the added links that connect the graphs, making them more like the E-R links than the S-F. If only the number of random links among graphs increases for fixed size graphs (Figure 5b blue circle marks), a maximum precision of about 0.9 is reached before degradation. That is, at around 32.5% of the links randomly connecting the components. In other words, the algorithm is robust, telling apart noise from structure even after up to 0.325 (from the random links connecting the components) + 0.5 (from the E-R component) = 0.825, i.e. 82.5% of all links are random. On the other hand, the number of false positives is constant at about 5%. In the case shown in Figure 5b, all of the false positives (red links not connecting the 2 graphs with different topologies) are inside the E-R graph and mostly non-existent on the side of the less random S-F graph.

Core Functions in the Wolfram Language

Both for Causal Deconvolution and for comparison against other methods (MI, NCD, and PID)

```
CausalDeconvolution[array_] :=
 Module[{pointrowmutation =
    Flatten[Table[
      ReplacePart[array, {{i, j} -> Mod[array[[i, j]] + 1, 2]}], {i,
       Length[array]}, {j, Length[array[[1]]]}], 1]},
  Reverse[SortBy[
    Thread[{Range[Length[pointrowmutation]],
      BDM[array, 4] - (N /@ BDM[#, 4] & /@ pointrowmutation)}],
    Last]]]

PIDMI[array_] :=
```

```
 Module[{pointrowmutation =
    Flatten[Table[
      ReplacePart[array, {{i, j} -> Mod[array[[i, j]] + 1, 2]}], {i,
       Length[array]}, {j, Length[array[[1]]]}], 1]},
  Reverse[SortBy[
    Thread[{Range[Length[pointrowmutation]],
      MutualInformation[#, array] & /@ pointrowmutation}], Last]]]

PIDNCD[array_] :=
 Module[{pointrowmutation =
    Flatten[Table[
      ReplacePart[array, {{i, j} -> Mod[array[[i, j]] + 1, 2]}], {i,
       Length[array]}, {j, Length[array[[1]]]}], 1]},
  Reverse[SortBy[
    Thread[{Range[Length[pointrowmutation]],
      NCD[#, array] & /@ pointrowmutation}], Last]]]

MutualInformation[x_, y_] :=
 N[Entropy[x] + Entropy[y] -
   Statistics`Library`NConditionalEntropy[x, y]]

NCD[x_, y_] :=
N@Block[{
    cx = ByteCount[Compress[x]],
    cy = ByteCount[Compress[y]],
    cxy = ByteCount[Compress[Join[x, y]]]
    },
   (cxy - Min[cx, cy])/Max[cx, cy]
   ]

CalculateInformationRowBDM[array_] :=
 Table[Table[
   Abs[StringBDM[
      StringJoin[ToString /@ Take[array[[i]], {m, m + 6}]]] -
     StringBDM[
      StringJoin[ToString /@ Take[array[[i]], {m + 7, m + 12}]]]], {m,
     1, Length[array[[i]]] - 12}], {i, Length[array]}]

CalculateInformationRowMI[array_] :=
 Table[Table[
   MutualInformation[Take[array[[i]], {m, m + 6}],
    Take[array[[i]], {m + 7, m + 12}]], {m, 1,
    Length[array[[i]]] - 12}], {i, Length[array]}]
```

```
CalculateInformationRowNCD[array_] :=
 Table[Table[
   NCD[Take[array[[i]], {m, m + 6}],
    Take[array[[i]], {m + 7, m + 12}]], {m, 1,
    Length[array[[i]]] - 12}], {i, Length[array]}]
```

The rest of the code and functions, e.g. CTM and BDM, are in the Wolfram Notebook available at `https://www.algorithmicdynamics.net/software.html` and reusable code in R is also available at `https://github.com/allgebrist/Causal-Deconvolution-of-Networks/`. A limited online implementation for small examples can also be found in `http://www.complexitycalculator.com/deconvolution`.

Supplementary Figures

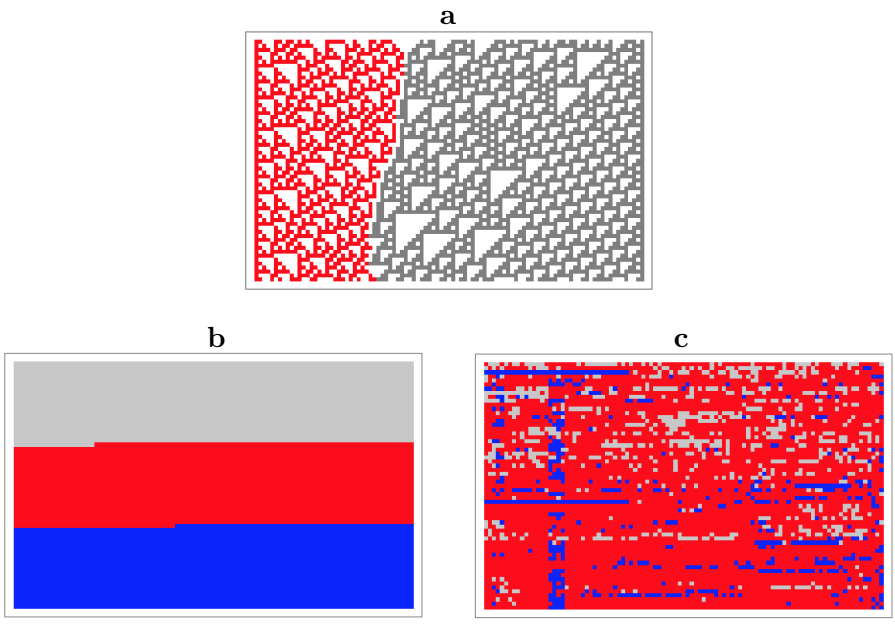

Figure 1: Comparison with other methods. a: Original interaction, b: Shannon entropy c: and lossless compression (Compress) all underperform, not being sensitive enough in performing the same task reported in Figs. 2c-f. Interacting rule 531441.

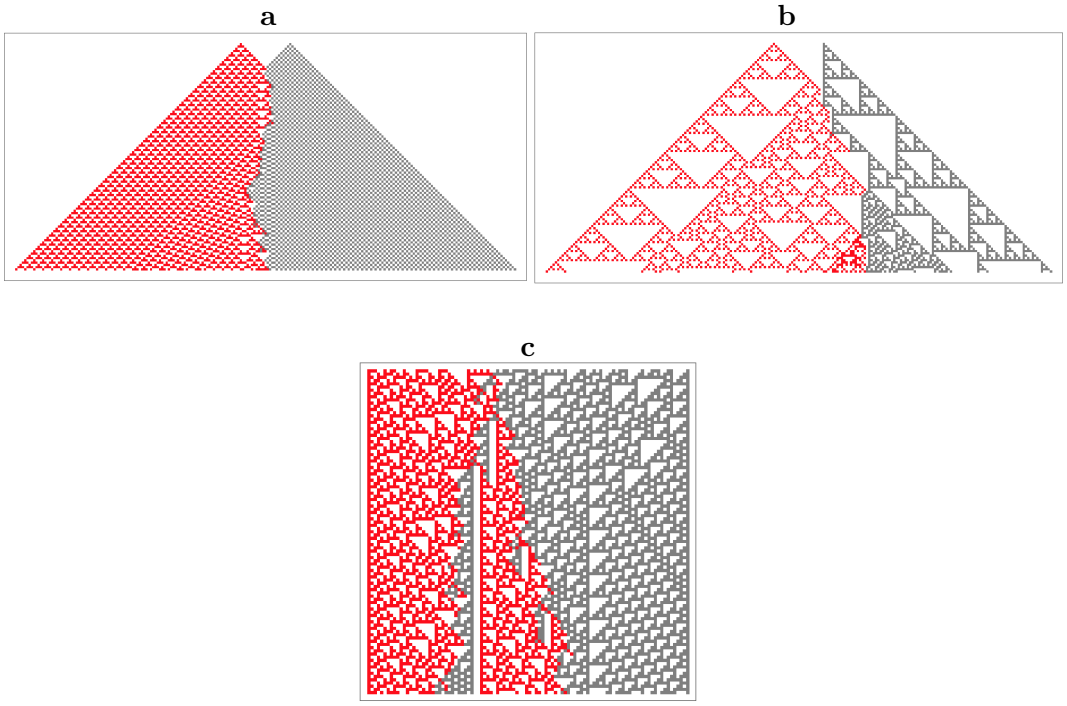

Figure 2: Examples of 2-system interactions using Elementary Cellular Automata (ECA), each running for 100 steps. a: Rules 54 (left) and 50 (right). b: Rules 82 (left) and 110 (right). c: Rules 60 (left) and 110 (right). In all cases, left and right rules can be seen to 'spill' into each others' space-time with non-trivial interacting rules. In all cases, the rule governing the interaction of the 2 ECA is rule with number 531441 according to the code provided in this Supplement

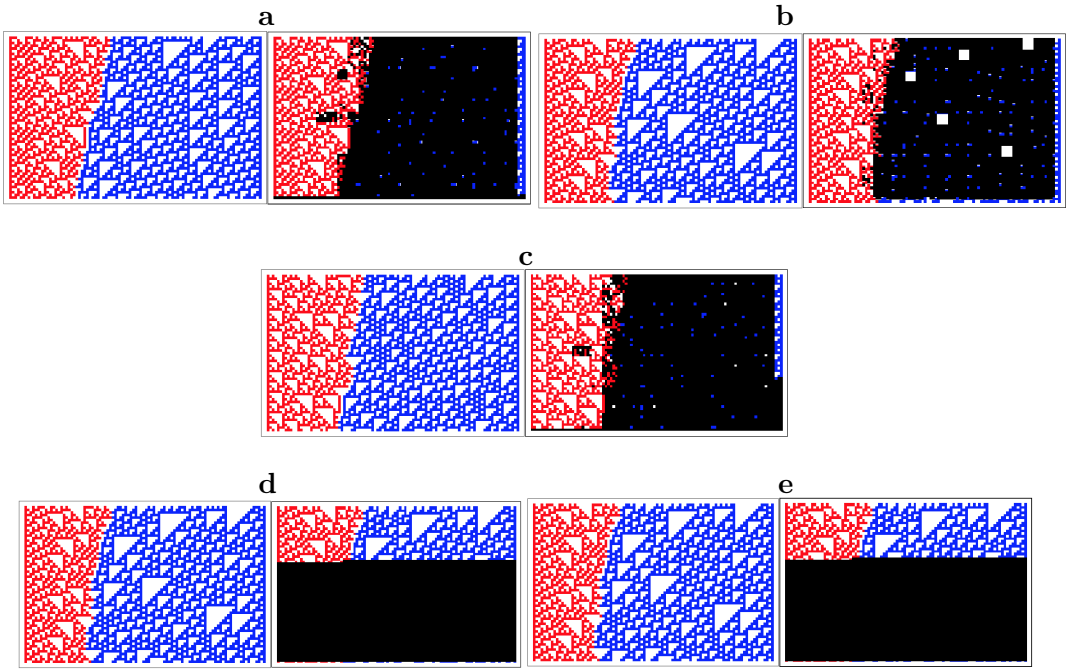

Figure 3: additional non-selected random cases of decomposition by causal deconvolution. a, b and c all involve rules 60 and 110 with non-trivial interaction with rule 531441 as introduced in this paper and with the same cutoff value. Each pair shows original (left) and decomposed (right) images on 3 different CA interacting cases. On the decomposed versions, black covers show the algorithm's success at isolating one region from another with high accuracy. d and e are implementations of 2 other methods based on a single case (b), but based on Partial Information Decomposition using (classical) Mutual Information and also Normalised Compression Distance. In this case, cutoff values produced a horizontal straight line in a random place, because the methods were unable to find any right-left separation stable enough to separate the image vertically.

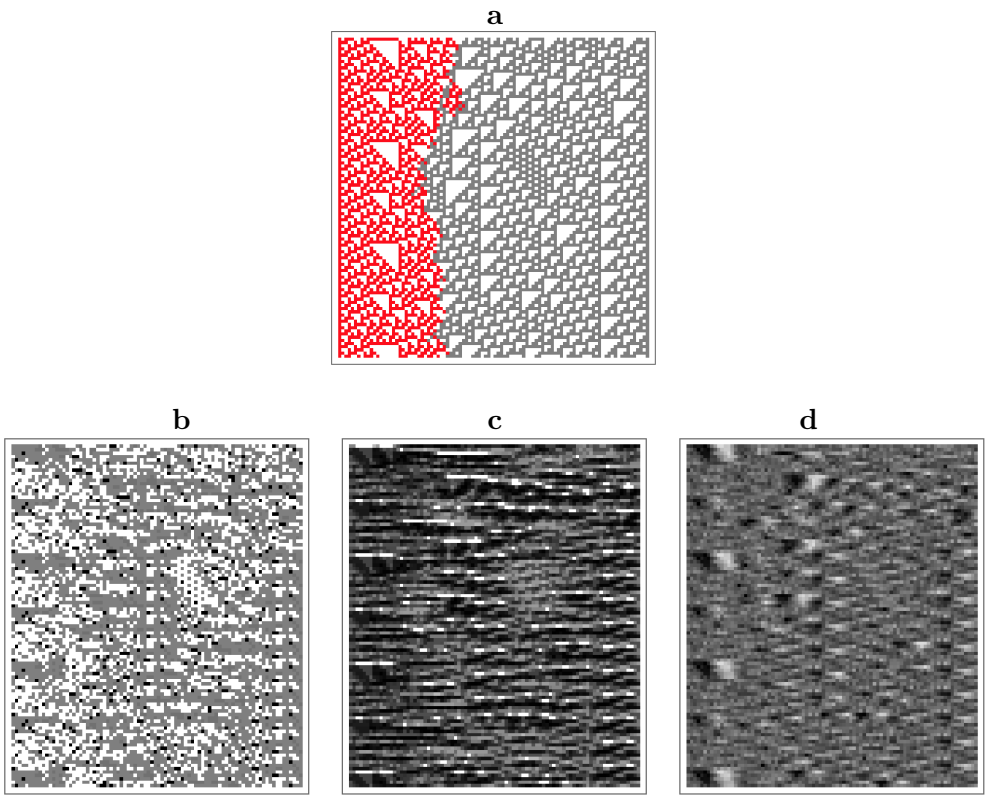

Figure 4: Sensitivity of methods applied to a 2-system interaction. a: both sides have similar qualitative properties. b: Result based on Partial Information Decomposition by Mutual Information, c: Normalised Compression Distance, and d: algorithmic causal deconvolution as introduced in this paper. These arrays are similar to redundancy lattices indicating either synergy (whiter) or redundancy (darker).

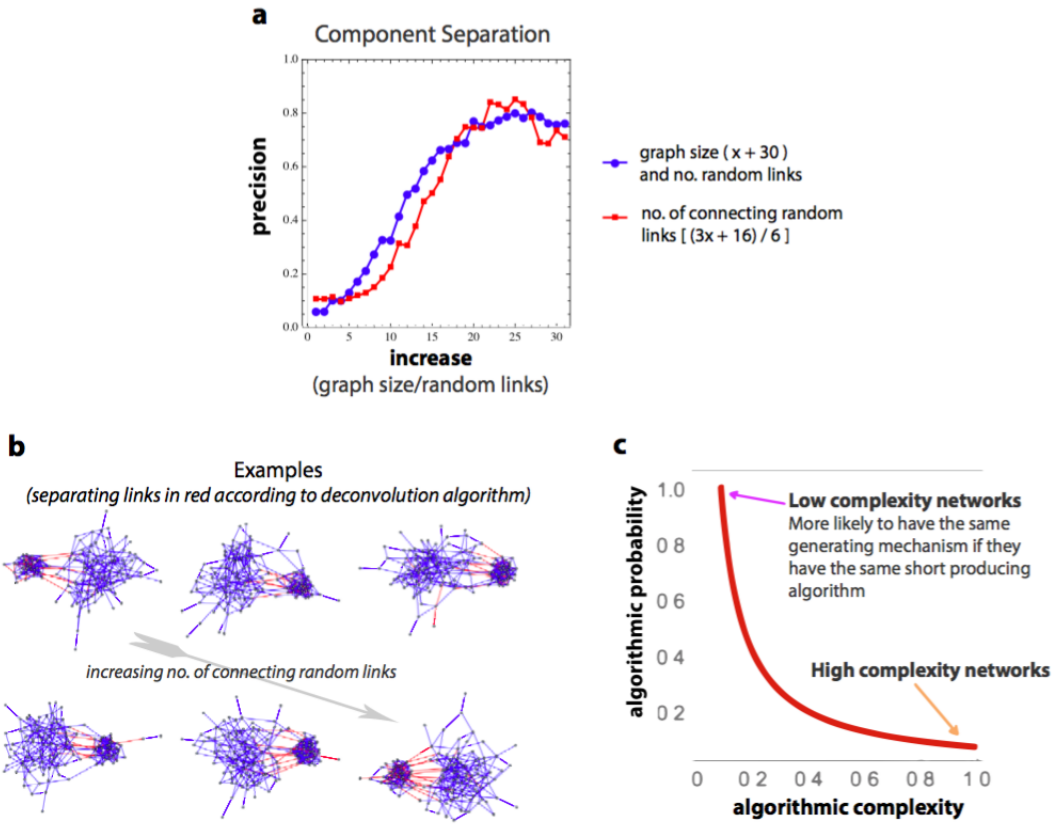

Figure 5: Robustness and limitations. a: Deconvolving an S-F graph generated by a preferential attachment algorithm (2 new links per added node) from an E-R (0.5 edge density) graph emulating noise (each point represents the average of 10 replicates). When the number of links increases as a function of the subgraph sizes, the separability is robust (red square markers) compared to increasing the number of random links only for graphs of fixed size (40 nodes), when successful separation is compromised, noise (E-R) and structure (S-F) being indistinguishable. b: A small sample of 6 graphs among the $10 \times 20 = 200$ graphs of S-F connected to E-R graphs considered in this study. Success at identifying randomly connecting links is shown and was found to be very robust. c: Objects have exponentially greater chances of being produced by the same generating mechanism if they are of low algorithmic randomness and thus of high algorithmic probability.

References

[1] Wolfram, S. *A New Kind of Science* (Wolfram Media, Champaign IL, 2002).

[2] Adams, A., Zenil, H., Reyes, E. & Joosten, J. The Effects of Global Rules on Interacting Cellular Automata. In J. Kari, I. Törmä, M. Szabados (eds.), *21st International Workshop on Cellular Automata and Discrete Complex Systems, AUTOMATA 2015*, TUCS Lecture Notes, 2015.

[3] Joosten, J. J. On the Necessity of Complexity. *Irreducibility and Computational Equivalence: 10 Years After the Publication of Wolfram's A New Kind of Science Ch. 2* (Springer, Berlin, 2013).

[4] Joosten, J. J. Complexity fits the fittest. *Complexity in Interdisciplinary Research and Applications Ch. 3* (Springer, Switzerland, 2013).

[5] Zenil, H. Compression-based Investigation of the Dynamical Properties of Cellular Automata and Other Systems. *Complex Systems* **19,** 1–28 (2010).

[6] Zenil, H. Asymptotic Behaviour and Ratios of Complexity in Cellular Automata Rule Spaces. *International Journal of Bifurcation and Chaos* **13,** 9 (2013).

[7] Zenil, H., Kiani, N. A. & Tegnér, J. Quantifying Loss of Information in Network-based Dimensionality Reduction Techniques. *Journal of Complex Networks* **4,** 342–362 (2016).

[8] Hermo-Reyes, E. & Joosten, J. J. "Competing Cellular Automata" `http://demonstrations.wolfram.com/CompetingCellularAutomata/`, Wolfram Demonstrations Project, June 17, 2014.

[9] Langton, C. G. Studying artificial life with cellular automata. *Physica D: Nonlinear Phenomena* **22,** 120–149 (1986).

[10] Zenil, H., Soler-Toscano, F., Dingle, K. & Louis, A. Graph Automorphisms and Topological Characterization of Complex Networks by Algorithmic Information Content. *Physica A: Statistical Mechanics and its Applications* **404,** 341–358 (2014).

From the main text:

References

[1] Zenil, H., Kiani, N.A., Marabita, F., Deng, Y., Elias, S., Schmidt, A., Ball, G. & Tegnér, J. An Algorithmic Information Calculus for

Causal Discovery and Reprogramming Systems, Available at SSRN: http://dx.doi.org/10.2139/ssrn.3193409 (2018).

[2] Zenil, H., Kiani, N.A., Zea, A.A., Rueda-Toicen, A. & Tegnér, J. Data Dimension Reduction and Network Sparsification Based on Minimal Algorithmic Information Loss. Preprint at https://arxiv.org/abs/1802.05843 (2018).

[3] Lloyd, S.P., Least squares quantization in PCM. *IEEE Transactions on Information Theory* **28,** 129–137 (1982).

[4] Kaufman, L. & Rousseeuw, P. J. Clustering by means of Medoids. *Statistical Data Analysis Based on the L_1-Norm and Related Methods* (North-Holland, Amsterdam, 1987).

[5] Ben-Hur, A., Horn, D., Siegelmann, H. & V.N. Vapnik, V. N. Support vector clustering. *Journal of Machine Learning Research* **2,** 125–137 (2001).

[6] Milo, R., Shen-Orr, S., Itzkovitz, S., Kashtan, N., Chklovskii, D. & Alon, U. Network Motifs: Simple Building Blocks of Complex Networks. *Science* **298,** 824–827 (2002).

[7] Newman, M.E. J. Finding community structure in networks using the eigenvectors of matrices. *Physyical Review E* **74,** 036104 (2006).

[8] Benczur, A. & Karger, D.R. Approximating s-t minimum cuts in $O(n^2)$-time. *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing (STOC'96)*, 47–55 (1996).

[9] Spielman, D.A. & Srivastava, N. Graph sparsification by effective resistances. *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing (STOC'08)*, 563–568 (2008).

[10] Spielman, D.A., Teng, S.-H. Spectral Sparsification of Graphs. *SIAM Journal of Computing* **40,** 981–1025 (2011).

[11] Liu, M., Liu, B. & Wei, F. Graphs determined by their (signless) Laplacian spectra. *Electronic Journal of Linear Algebra* **22,** 112–124 (2011).

[12] Granger, C. W.J. Investigating causal relations by econometric models and cross-spectral methods. *Econometrica* **37,** 424–438 (1969).

[13] Schreiber, T. Measuring Information Transfer. *Physical Review Letters* **85,** 461–464 (2000).

[14] Pearl, J. *Causality: models, reasoning and inference* (Cambridge University Press, Cambridge, 2000).

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

[16] Watanabe, S. Pattern recognition as information compression. *Frontiers of Pattern Recognition* (Academic Press, New York, 1972).

[17] Williams, P.L. & Beer, R.D. Nonnegative Decomposition of Multivariate Information. Preprint at https://arxiv.org/abs/1004.2515 (2010).

[18] Lizier, J.T., Bertschinger, N., Jost, J. & Wibral, M. Information Decomposition of Target Effects from Multi-Source Interactions: Perspectives on Previous, Current and Future Work. *Entropy* **20,** 307 (2018).

[19] Li, M. & Vitányi, P.M.B. *An Introduction to Kolmogorov Complexity and Its Applications 3ed* (Springer, New York, 2009).

[20] Li, M., Chen, X., Li, X., Ma, B. & Vitányi, P.M.B. The similarity metric. *IEEE Transactions on Information Theory* **50,** 3250–3264 (2004).

[21] Bennett, C. H., Gács, P., Li, M., Vitányi, P.M.B. & Zurek, W. H. Information Distance. *IEEE Transactions on Information Theory* **44,** 1407–1423 (1998).

[22] Cilibrasi, R. & Vitanyi, P.M.B. Clustering by compression. *IEEE Transactions on Information Theory* **51,** 1523–1545 (2005).

[23] Shannon, C.E. A mathematical theory of communication. *Bell System Technical Journal* **27,** 379–423 (1948).

[24] Ince, R.A.A. Measuring Multivariate Redundant Information with Pointwise Common Change in Surprisal. *Entropy* **19,** 318 (2017).

[25] Strelioff, C.C. & Crutchfield J.P. Bayesian Structural Inference for Hidden Processes. *Physical Review E* **89,** 042119 (2014).

[26] Shalizi, C.R. & Crutchfield, J.P. Computational Mechanics: Pattern and Prediction, Structure and Simplicity. *Journal of Statistical Physics* **104,** 819–881 (2001).

[27] Delahaye, J.-P. & Zenil, H. Numerical Evaluation of the Complexity of Short Strings: A Glance Into the Innermost Structure of Algorithmic Randomness. *Applied Mathematics and Computation* **219,** 63–77 (2012).

[28] Soler-Toscano, F., Zenil, H., Delahaye, J.-P. & Gauvrit, N. Calculating Kolmogorov Complexity from the Frequency Output Distributions of Small Turing Machines. *PLoS One* **9,** e96223 (2014).

[29] Hutter, M. Universal Artificial Intelligence: Sequential Decisions Based On Algorithmic Probability. *Texts in Theoretical Computer Science. An EATCS Series* (Springer, Berlin, 2005).

[30] Gauvrit, N., Zenil, H. & Tegnér, J. The Information-theoretic and Algorithmic Approach to Human, Animal and Artificial Cognition. *Representation and reality: Humans, animals and machines 117–139* (Springer, Berlin, 2017).

[31] Rissanen, J. Modeling by shortest data description. *Automatica* **14,** 465–658 (1978).

[32] Levin, L.A. Universal search problems. *Problems of Information Transmission* **9,** 265–266 (1973.

[33] Schmidhuber, J. The speed prior: a new simplicity measure yielding, near-optimal computable predictions. *Proceedings of the 15th annual conference on computational learning theory (COLT 2002) 216–228* (Springer, Sydney, 2002).

[34] Daley, R.P. Minimal-program complexity of pseudo-recursive and pseudo-random sequences. *Mathematical systems theory* **9,** 83–94 (1975).

[35] Zenil, H., Badillo, L., Hernández-Orozco, S. & Hernández-Quiroz, F. Coding-theorem Like Behaviour and Emergence of the Universal Distribution from Resource-bounded Algorithmic Probability, *International Journal of Parallel Emergent and Distributed Systems* (2018).

[36] Hernández-Orallo, J. Computational measures of information gain and reinforcement in inference processes. *AI Communications* **13,** 49–50 (2000).

[37] Hernández-Orallo, J. Universal and cognitive notions of part. *Proceedings of 4th Systems Science European Congress*, 711-722 (1999).

[38] Solomonoff, R.J. The Time Scale of Artificial Intelligence: Reflections on Social Effects, *Human Systems Management* **5,** 149–153 (1985).

[39] Zenil, H., Hernández-Orozco, S., Kiani, N.A., Soler-Toscano, F., Rueda-Toicen, A., & Tegnér, J. A Decomposition Method for Global Evaluation of Shannon Entropy and Local Estimations of Algorithmic Complexity. *Entropy* **20,** 605 (2018).

[40] Chaitin, G. J. On the length of programs for computing finite binary sequences. *Journal of the ACM* **13,** 547–569 (1966).

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

[42] Zenil, H., Kiani, N.A. & Tegnér, J. Symmetry and Correspondence of Algorithmic Complexity over Geometric, Spatial and Topological Representations. *Entropy* **20,** 534 (2018).

[43] Zenil, H., Soler-Toscano, F., Delahaye, J.-P. & Gauvrit, N. Two-Dimensional Kolmogorov Complexity and Validation of the Coding Theorem Method by Compressibility. *PeerJ Computer Science* **1,** e23 (2013).

[44] Riedel, J. & Zenil, H. Rule Primality and Compositional Emergence of Turing-universality from Elementary Cellular Automata. *Journal of Cellular Automata*, vol. 13, pp. 479497, 2018.

[45] Hernández-Orozco, S., Kiani, N.A. & Zenil, H. Algorithmically probable mutations reproduce aspects of evolution, such as convergence rate, genetic memory, and modularity. *Royal Society Open Science* **5,** 180399 (2018).

[46] Albert, R. & Barabási, A.-L. Statistical mechanics of complex networks. *Reviews of Modern Physics* **74,** 47–97 (2002).

[47] Gulwani, S., Hernández-Orallo, J., Kitzelmann, E., Muggleton, S. H., Schmid, U. & Zorn, B. Inductive programming meets the real world. *Communications of the ACM* **58,** 90–99 (2015).

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

[49] Zenil, H., Kiani, N.A. & Tegnér, J. Methods of Information Theory and Algorithmic Complexity for Network Biology. *Seminars in Cell and Developmental Biology* **51,** 32–43 (2016).

[50] Zenil, H., Kiani, N.A. & Tegnér, J. Low-Algorithmic-Complexity Entropy-deceiving Graphs. *Physical Review E* **96,** 012308 (2017).