Real patterns and indispensability

Abel Suñé https://orcid.org/0000-0003-3013-5208
Department of Philosophy – Universitat de Barcelona
LOGOS Research Group in Analytic Philosophy
Barcelona Institute of Analytic Philosophy
Montalegre, 6. 08001. Barcelona (Spain)
martinezsune@ub.edu

Manolo Martínez https://orcid.org/0000-0002-6194-7121
Department of Philosophy – Universitat de Barcelona
LOGOS Research Group in Analytic Philosophy
Barcelona Institute of Analytic Philosophy
Montalegre, 6. 08001. Barcelona (Spain)
mail@manolomartinez.net

Abstract

While scientific inquiry crucially relies on the extraction of patterns from data, we still have a very imperfect understanding of the metaphysics of patterns—and, in particular, of what it is that makes a pattern real. In this paper we derive a criterion of real-patternhood from the notion of conditional Kolmogorov complexity. The resulting account belongs in the philosophical tradition, initiated by Dennett (1991), that links real-patternhood to data compressibility, but is simpler and formally more perspicuous than other proposals defended heretofore in the literature. It also successfully enforces a non-redundancy principle, suggested by Ladyman and Ross (2007), that aims at excluding as real those patterns that can be ignored without loss of information about the target dataset, and which their own account fails to enforce.

Acknowledgements

1   Introduction

Scientific inquiry often depends on the extraction of patterns from data. The huge datasets and corpora typical of many contemporary scientific projects have only made this dependence more obvious and central. Genomics (Barrett et al. 2013), connectomics (Sporns, Tononi, & Kötter 2005), and astronomy (Feigelson and Babu 2012) are obvious examples, but the trend is quite general. Philosophers of science have been paying increasing attention to patterns, so as to keep up with the trend, but also in the hope that these entities will fruitfully supplement other entities, such as natural kinds, in the metaphysics of science (Andersen 2017; Ladyman and Ross 2007; Martínez 2015, Martínez 2017; McAllister 2003b, 2003a, 2011; Petersen 2013).

On the other hand, as is often the case when a new theoretical tool starts gaining prominence, we still have a very imperfect understanding of the notion of pattern itself. In this paper we propose a way to spell it out that builds upon Ladyman and Ross's theory of *real patterns* [RP henceforth], the most sophisticated account currently on offer. RP substantially extends and refines the idea (first put forward in Dennett (1991) and prefigured, more or less explicitly, by Bogen & Woodward (1988), and Rissanen (1998), among many others) that there are patterns present in a dataset D insofar as one can describe a computer program that outputs D while being shorter than D—a program, that is, that *compresses* D. The underlying insight is that patterns correspond to redundancies in the dataset, and it is these redundancies that are exploited by the algorithm implemented by the program.

A further question is what, precisely, is a pattern. The "underlying insight" just mentioned does not take a stand on this: we are invited to conclude that there are patterns present in D if D is compressible, but we have been given no guidance, for any entity P, as to whether it is warranted to claim that P is a pattern in D. This is the question that Don Ross and James Ladyman take up in a series of papers culminating with their 2007 book, and which result in their RP account of real patterns. While RP represents substantial progress toward the development of a metaphysics of patterns, somewhat surprisingly its concrete formulation has not been closely scrutinized in the literature—perhaps, we speculate, because it builds on a computer-theoretic toolbox that is comparatively alien to many metaphysicians of

science.

Close scrutiny shows that RP is not without problems: one of the goals of the theory is to provide a criterion of *non-redundancy* [also, interchangeably, *indispensability*] for patterns—the idea, roughly, being that one should count as real all and only those patterns such that ignoring them results in an incomplete description of the world. We will show that RP does not successfully provide such a criterion. We will also offer an alternative definition of real pattern, simpler and more formally perspicuous than RP, that provides a workable criterion of indispensability.

Before getting on to this, though, it is important to clarify the scope of the discussion in this paper. In the literature we are dialoguing with, 'real' is taken with a grain of salt: e.g., Dennett refuses to answer the question whether his account of real patterns is «a sort of instrumentalism or a sort of realism» (1991, p. 51), and Ladyman and Ross [also L&R henceforth] explicitly endorse «a version of instrumentalism about all propositions referring to self-subsistent individual objects, chairs and electrons alike» but also «realism about the domain of scientific description» (2007, p. 198). For our current purposes, this is the sort of realism that matters. We are currently interested in developing a coherent notion of real pattern, in the comparatively soft understanding of realism typical of the literature on real patterns: we are after a principled way to ground the claim that some patterns are, but some other patterns are not, present in a certain dataset. Whether the criterion for real-patternhood to be developed here is also able to underpin stronger versions of scientific realism (having to do, for example, with the existence of certain objects or kinds of objects) is an extremely interesting topic that will have to be taken up on another occasion.

A point on terminology: L&R use "pattern" to refer to arbitrary strings of symbols, independently of whether they can be used to compress a dataset or not (see fn. 6). We will use "string" for this purpose. In this paper we reserve "pattern" for the Dennettian notion of an aspect of a dataset that makes it compressible. Finally, we will follow L&R in using "real pattern" to refer to non-redundant (Dennettian) patterns, where non-redundancy is understood as above.[1]

In section 2 we introduce Dennett's original insight, and the computer-theoretic notions on which it builds. In section 3 we explain why current approaches to model selection in algorithmic information theory don't tell the whole story about patterns. In section 4 we summarize Ladyman and Ross's RP account, and then, in section 5, we present an

---

[1]   The above informal characterization of non-redundancy will be sharpened in section 4.

important shortcoming of RP: we describe a model that shows that it does not abide by their own indispensability principle—sketched above, and more fully in section 4. Finally, in section 6 we put forward a better definition of real pattern, based on the notion of conditional Kolmogorov complexity, that successfully incorporates an indispensability principle. Section 7 offers some concluding remarks.

2   Algorithmic Complexity and Patterns

Philosophical inquiry into the role and nature of patterns in science kicks off with Daniel Dennett's seminal 'Real Patterns' (1991). Dennett's main insight is that patternhood is linked to *algorithmic compression.* This is just the compression contemporary everyday life has made us familiar with: FLAC sound files or TIFF image files are *compressed*, in the sense that these files are shorter than the uncompressed wav or bitmap originals. Speech transmitted wirelessly through cell phone networks is also heavily compressed (Rappaport 1996). What all instances of algorithmic compression have in common, regardless of their medium or the object of compression, is the fact that a target object is faithfully reproduced by something shorter than a full bit by bit description.[2]

For a more explicit example, consider the following two objects: the first object is a list of results in a series of one million tosses of a fair coin (1 encodes heads, and 0 tails): 0100011011...; the second object is the following list of one million binary digits: 010101... ...010101. Since we are dealing with a fair coin, the first object is a random string of 0s and 1s. The second string, however, involves an obvious pattern: it's a repetition of '01' half a million times. In order to describe the first string there are no substantial shortcuts to writing, digit by digit, the results of each coin toss. On the other hand, the second string can be fully reproduced via an algorithmic description that is much shorter than the full string: (roughly) "print '01' half a million times". The first, *random* string is incompressible. The second, *patterned* one incorporates redundancies that can be exploited by a compression algorithm.

Chaitin (1966), Kolmogorov (1965) and Solomonoff (1964a, 1964b), the founders of algorithmic information theory, suggested independently that a sequence should be considered random if and only if it is incompressible. Dennett conjoined this thesis with the

---

[2]   Speech compression is an instance of *lossy* compression, where faithfulness of compression is judged by a certain distortion measure, or loss function (Cover & Thomas, 2006, ch. 10; Shannon, 1959). The main notion of compression we rely on in what follows, on the other hand, is *lossless* compression, in which the original file and the decoded version thereof are identical. TIFF, FLAC and others such as DEFLATE, typically used in zip files, are widely popular lossless algorithms.

We note that the very existence of lossless compression algorithms appears to be in some tension with McAllister's (2003a) claim that empirical datasets are incompressible—insofar, e.g., as empirical datasets can contain photographs or audio recordings. We won't pursue this topic here.

idea that a string has patterns to the extent that it is not purely random: "a [string] is not random – has a pattern -- if and only if there is some more efficient way of describing it" (1991, p. 32). In a nutshell, *a pattern in a dataset is any aspect in the dataset that allows for it to be compressible.*

As we explained above, compressibility in a string is to be thought of as affording the existence of comparatively short programs that can output the string in question.[3] For some strings there will be multiple programs of different lengths that can output them, but we will be interested only in the shortest such program. The *maximum achievable algorithmic compression*, or *Kolmogorov Complexity*, K(S), of a given string, S, is *the length of the shortest program that outputs S*. The Kolmogorov complexity of strings will be widely appealed to in what follows.

A related notion is the *complexity of a string,* S*, conditional on another string,* T*, or conditional Kolmogorov complexity* K(S|T). This is the length of the shortest program that outputs S, if it is allowed to use T as an input. Suppose, for example, that we have to compress a string, S, that encodes a recording of a spoken conversation. The length of the shortest program that outputs S (i.e., the program that represents the best possible compression of S) gives K(S). But suppose as well that we have a computer library, T, that encodes certain speech statistics typical of recordings such as S. This will often mean that we can write a shorter program that prints S, if we rely on T as an additional input to the program. The shortest such program gives K(S|T).

Finally, one can quantify the amount of information an object x carries about another object y—the *mutual information* between x and y—as a measure of the reduction in the amount of descriptive effort one has to make to describe y after knowing x. More formally, if S and T are strings, the algorithmic mutual information between them, I(S:T), is:

I(S:T) = K(S) – K(S|T). (Li & Vitányi 2008, definition 3.9.2)

---

[3]   In formal presentations of Kolmogorov complexity (e.g., Li & Vitányi, 2008, p. 107), the programs we have been alluding to are inputs to a reference universal Turing machine [UTM] . For the purposes of this paper, we can just think of the reference UTM as implementing one of the very many popular Turing-complete programming languages⸺say, Python, or Javascript.

Petersen, (2018, p. 2) discusses whether the choice of UTM introduces a bias in the resulting account of patterns (for example, by making any arbitrary dataset, however big and random, compressible and hence patterned) and concludes, with Li & Vitányi, (2008, p. 112), that a small enough UTM will make any such potential bias negligible.

3 Model Selection is not Pattern Individuation

We claimed above that Dennett's foundational insight was that a pattern in some dataset is any aspect in it that allows for it to be compressible. "Any aspect" is, of course, rather unspecific. A full theory of patterns, if it is to help us to recognize and individuate patterns, needs to spell out in more detail what these compression-enabling aspects of datasets amount to. In the following section we discuss Ladyman and Ross's attempt to provide this detail (2007; Ross 2000). Before, in this section, we briefly discuss an approach in algorithmic information theory to a related question, and explain why this approach does not deliver an account of patterns, at least given the way metaphysicians of science employ this notion.
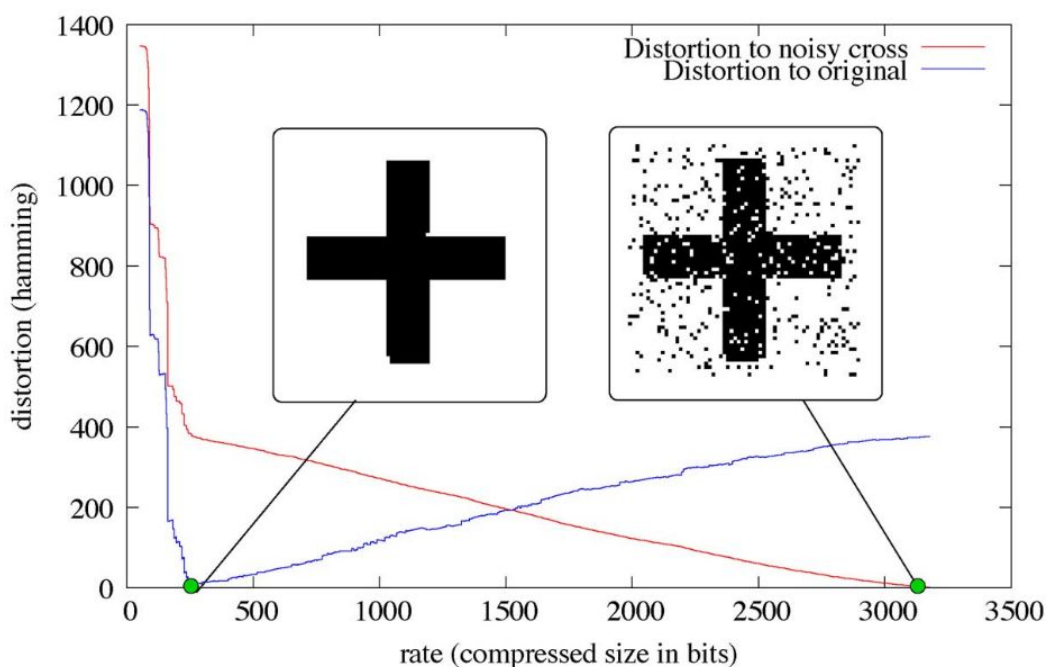

Figure 1. Denoising based on structure functions (from Vereshchagin and Vitanyi 2010)

Consider a noisy image; e.g., the right cross in figure 1 (reproduced from Vereshchagin & Vitanyi 2010, p. 3446). We can intuitively analyze this object into two components: first «the information accounting for the useful regularity present in the object» (Vitányi 2006, p. 4617), that captures the noiseless cross to its left; and a meaningless one, «the information accounting for the remaining accidental information» (*ibid*.), that captures the noise.

The way algorithmic information theory approaches this analysis (following suggestions made by Kolmogorov in the early 70s and in his 1965) is by devising ways to encode the object in which the relevant code has two parts: the first part captures the meaningful

information of the object (i.e., provides a *model* of the object); the second part, the noisy remainder. In the foundational, «structure functions» version of this idea (Vereshchagin & Vitányi 2004, 2010), this two-part code is implemented in the following way. To encode a string x, we first identify a set S such that x is a *typical* member of it. Being a typical member of a set simply means that, in order to pick out x among other members of S, there's no shorter procedure than giving its position in an arbitrary enumeration of S's members. A two-part code, then, can be constructed that first reconstructs S (this takes K(S) bits, by the definition of Kolmogorov complexity), and then gives x's position in S (this takes log|S| bits, where |S| is S's size). S corresponds to the best model of the data; x's position in S, to the noisy remainder.

It can be shown (Vereshchagin & Vitányi, 2004, p. 3269) that for each string x there are *optimal sets* for which the sum K(Soptimal) + log|Soptimal| is equal to K(x). That is, somewhat surprisingly, the K(Soptimal) term (which captures the meaningful information in x by identifying the simplest set in which x is a typical member) together with the log|Soptimal| term (which captures the noisy remainder by picking out x in a brute enumeration) add up to the Kolmogorov complexity of the original string (up to an additive constant). The best meaning-plus-noise code for x is as good as the best possible code for x.[4]

While algorithmic model selection is designed to help us set apart meaningful from meaningless in a dataset, there are at least two respects in which it does not provide a solution to the problem real-pattern theorists are interested in. First, the two-part code idea aims at reconstructing the original string x *in its entirety*. In our case, x would correspond to an empirical dataset and applying the procedure just sketched would leave us with a specification of *all meaningful regularities* in the dataset together with the remaining noise. But the virtual totality of patterns identified in actual scientific practice correspond to *partial* regularities in the target dataset. For one example among very many, take *CpG islands,* areas of DNA with high concentration of the CpG dinucleotide. The abundance of CpG in a certain stretch of DNA is a clear pattern, widely studied in epigenetics (Bird 1986; Larsen, Gundersen, Lopez, & Prydz 1992). Yet, of course, full knowledge of where CpG islands are, on its own, does not allow us to reconstruct a full genome. In general, patterns in a dataset illuminate important aspects of it, without fully describing it. Model selection in the algorithmic information-theoretic tradition, as described above, offers no guidance as to how to uncover or describe patterns in this sense.

---

[4]   The foregoing few paragraphs only scratch the surface of the algorithmic approach to model selection. This is the aim of so-called *algorithmic statistics*. We point the interested reader to Gács, Tromp, & Vitányi (2001), Vereshchagin & Shen, (2017) and references therein for in-depth discussion and alternatives to the structure-functions two-part code.

The first problem we have mentioned with the model selection approach is that it offers a way to identify *all* meaningful information in a dataset, in one go, but not a way to identify partial, incomplete portions of this information, which is what patterns are.[5] The second respect in which model selection is not the right tool for pattern discovery stems from the fact that many of the patterns identified by scientists are partially constituted by what, arguably, is noise by model-selection lights. Again, for one example among many, the DNA of the *C. elegans* worm contains non-coding regions of 10-base pair periodic adenine [A] / thymine [T]-clusters (or PATCs). These are regions of non-coding DNA in which stretches of a few consecutive As and stretches of a few consecutive Ts happen more or less every ten bases, and they make up approximately 10% of the C. elegans genome (Frøkjær-Jensen et al., 2016). PATCs have been described as important patterns in the *C. elegans* genome, as they appear to play a role in allowing germline expression of transgenes, in regions the expression of which is otherwise silenced (*ibid.*). The fact that the clusters in PATCs happen every 10 bases (and not 20 or 40), and that the bases involved are A and T (and not C or G) is likely to be random happenstance to a certain degree—that is, noise, in model-selection parlance. But it is PATCs themselves, their noisy ingredients included, and not just the «meaningful» core identified by structure functions, that are relevant to *C. elegans* genetics.

An account that accommodates partial patterns (such as CpG islands), and noise-including patterns (such as PATCs) is thus in order. Before presenting our own, we turn now to describing the most developed, if ultimately unsuccessful, such account.

4   Ladyman and Ross' Real Patterns Theory[6]

L&R's main idea is that the «aspects» of datasets that enable compression can be captured by identifying strings that partially encode the original dataset—these strings will be the patterns in the dataset. We will be presently more precise than this; but we can already note that there can be many different strings that partially encode a dataset—indeed, there can be sets of *mutually redundant* strings, in the sense that each of them informs us of the very same aspects of the target dataset.

L&R contend that the right theory of *real* patterns should provide guidance in the process of

---

[5]   To be clear: this is a problem insofar as we want to use model selection as a method to identify patterns. Model selection is a perfectly clear goal in algorithmic statistics, and the structure-function approach has much to recommend it, for its intended purpose.

[6]   We are greatly indebted to an anonymous referee for very generous and detailed input that has much improved this section.

choosing which members to recognize as real patterns, out of those possible sets of mutually redundant strings. That is, a theory of patterns must help us distinguish between potentially useful but ultimately dispensable patterns, which one can ignore without any ontological loss, and patterns such that ignoring them results in an incomplete description of the target dataset (*cf.* Ladyman and Ross 2007, p. 231):

> *Non-redundancy principle*: Include in your ontology all and only those patterns that are *required* for a full (lossless) reconstruction of the target dataset.

L&R define real patterns as follows:

> A pattern[7] P is real iff
> (i) it is projectible; and
> (ii) it has a model that carries information about at least one pattern D in an encoding that has logical depth less than the bit-map encoding of D, and where D is not projectible by a physically possible device computing information about another real pattern of lower logical depth than P (Ladyman and Ross 2007, p. 233)[8]

This definition uses a number of comparatively uncommon technical terms. Our first aim in this section will be to present a version of L&R's theory of real patterns that captures the main gist of their original definition but is both simpler and more continuous with the rest of the literature on patterns. We will now discuss each of the technical terms in turn.

Projectibility

Projectibility is used twice in the definition. L&R say that an *entity x projects an entity y* (notated x→y) iff it is possible to calculate y from x (*ibid.*, p. 224). Take, for example, a system of physical bodies moving about in space. We might want to predict future positions and velocities of one particular object given data about the current positions and velocities of a range of objects. Let x be a specification of the position and velocity of those objects at time t; and let y be a specification of the position and velocity of our target object at a later point in time. If it is possible to effect a computation x→y which outputs y when input x (by, say, solving differential equations that correspond to some deterministic theory of gravity) we

---

[7]   Keep in mind that patterns *simpliciter* for L&R are just what we have called "strings". This is what fn. 51 in Ladyman and Ross (2007) amounts to: "A mere pattern is a locatable address associated with no projectible or non-redundant object." (*ibid.*, p. 231). See also *ibid.,* p. 229: "From the ontological point of view, a non-projectible pattern exactly resembles the traditional philosophical individual.". For the meaning of *locator* and cognates in L&R's system, see *ibid.,* p. 121ff. For the related notion of "perspective" see *ibid.,* p. 224.

[8]   We have changed variable names to align them with the ones we use in this paper.

say that x projects y. If an object x projects an object y, x doesn't merely carry some amount of information about y: it carries *all the information necessary to specify y without residue*. In algorithmic complexity terms: for any strings x and y, x projects y iff $K(y|x) = 0$, up to an additive constant independent of x and y.

According to L&R, a pattern (e.g., a differential equation as in the example above) that makes possible a projection, x→y, in the sense explained, is *projectible*---as in clause (i) above---in case it affords projecting ys for multiple unobserved input xs. As L&R point out, this condition aims to «avoid trivialization of projectibility by reference to [a computer] that simply implements the one-step inference rule 'Given input [x], output [y]'» (*ibid*.). Yet it could be argued that the appeal to multiple unobserved inputs doesn't avoid trivialization: projecting multiple ys for multiple xs is computationally equivalent to projecting a single, bigger y (say, an array of the original ys) from a single, bigger x (an array of the original xs). Again here, there is a Turing machine that calculates the bigger y from the bigger x by (paraphrasing L&R) "simply implementing the one-step inference rule 'Given input [bigger x], output [bigger y]".

In our paper we are, in effect, taking «y is projectible from x» to mean «a trivial (very short, etc.) universal Turing machine running x as its program outputs y». This seems to capture what L&R want projectibility to do, while avoiding triviality. In a Kolmogorov complexity setting these constraints on universal Turing machines are standard (see fn. 2).

Models

L&R's definition also makes reference to *models* of patterns. This has two functions in L&R's construction: first, it helps make explicit that, in order to apply algorithmic complexity theory to real-world phenomena, we need to translate them to strings. Second, it makes the definition applicable to cases in which we «may have access only to a model of the pattern in question» (e.g., when the pattern is the interior of the Sun, *ibid.* p. 233). The counterexample to RP that we will be considering in the sequel is formulated directly in terms of strings. This will allow us to sidestep these complications.

Encodings

As we saw above, there are patterns in datasets iff the latter can be compressed. The uncompressed, raw version of a dataset (for example, the raw lists of numbers and text that result from research) is what L&R call its *bit-map encoding* (see *ibid.,* p. 232). This is the

string we are calling "D" in this paper. Other, more sophisticated encodings of the dataset might provide more economical (shorter) representations of it, if it is compressible.

Logical depth

Finally, the logical depth of a string is the number of steps necessary to output it from its minimal program (Bennett 1995). John Collier, co-author with Ladyman and Ross of the main chapter on real patterns in Ladyman and Ross (2007), has given reasons to opt for logical depth when discussing physical complexity in (Collier 2001; Collier & Hooker, 1999). In our context, this is a relatively idiosyncratic choice: it is universally understood that the main measure of algorithmic complexity is Kolmogorov complexity (Grünwald and Vitanyi 2008) and this is also the notion used in most other papers in the philosophical literature on real patterns (Dennett 1991; McAllister 2003b; Petersen 2013, 2018). We will opt for a Kolmogorov-complexity version of L&R's definition of real patterns. In any event, the examples we will be using to make our points in this paper are, in Bennett's sense, *logically shallow*. That is, for them Kolmogorov complexity equals logical depth (and, indeed, mostly equals length). Still, if one wishes to stick to logical depth, this is perfectly doable. Where we say "Kolmogorov complexity" one should read "logical depth", and where we say "conditional Kolmogorov complexity" one should read "relative logical depth" (Bennett 1995, definition 1.1).

With these clarifications in mind we can present a succinct version of L&R's definition:

> **RP:** A pattern P is a real iff there is a dataset D such that
> (i) I(D:P) > 0; and
> (ii) there is no Q such that K(D|Q) = 0 and K(Q) < K(P).

The definition is, first, saying that for a string P to be a real pattern it has to carry information about some dataset D—this in turn means that a program that has P as input and outputs D can be shorter than a program that outputs D and has no inputs. This captures the Dennettian compressibility condition on patternhood. Second, the definition is saying that if P is a real pattern, then no pattern shorter than P has all the information needed to specify D without residue. This attempts to enforce the non-redundancy principle above: if there was a pattern that losslessly compressed D as a whole, while being shorter than P, P would indeed be superfluous for the purposes of describing D.

There is an asymmetry in the way this definition handles the Dennettian and the non-redundancy ingredients: on the one hand it deems it enough that P *carry some, not all,*

information about D; yet, on the other hand, redundancy is averted by appealing to *projectibility*—which, recall, requires for a pattern that it be *fully* informative about D, if it is to make P redundant. This combination is an understandable theoretical choice. First, non-redundancy cannot be implemented in terms of information carrying: if the second part of the definition merely read

(ii)* there is no Q such that K(D|Q) < K(D) and K(Q) < K(P)

the unreasonable consequence would be that patterns less complex than P that carried *less* information about D than P would make P redundant. Second, and conversely, for the sort of reasons we saw in the previous section, the Dennettian insight cannot be implemented in terms of projection: it is unreasonable to rule that the only real patterns in a dataset are those patterns that *fully* reconstruct the dataset (recall CpG islands).

This combination is an understandable theoretical choice, but it is the wrong one: as we show in the following section, the interplay between projectibility and information-carrying in clause (ii) of RP doesn't quite behave in the manner intended, and this can be used to smuggle in very noisy patterns, and have them qualified as real.

5. The Problem

In RP, putatively real patterns are tested by how well they explain a dataset D. We present a toy model in which D is a string constructed in the following way: we first take three 200MB[9] random binary strings, A1, A2, A3. The concatenation of these three strings we call A. Second, we construct D as the concatenation of two exact copies of A. The size of D, thus, is (200 × 3) × 2 = 1200MB (see Fig. 2). Alongside D, we will construct another pattern, Q, as the concatenation of A1 and an extra 200MB of random binary digits—A1 plus a lot of superfluous noise, that is.[10]

---

[9]   A megabyte (MB) is one million bytes.

[10]   For ease of discussion, we have designed our example so that redundancies in the dataset are readily apparent. Of course, in more realistic examples, sophisticated coding might be needed in order to squeeze the redundant material out of our target string. The kind of argument we develop here applies to the more realistic cases as well.
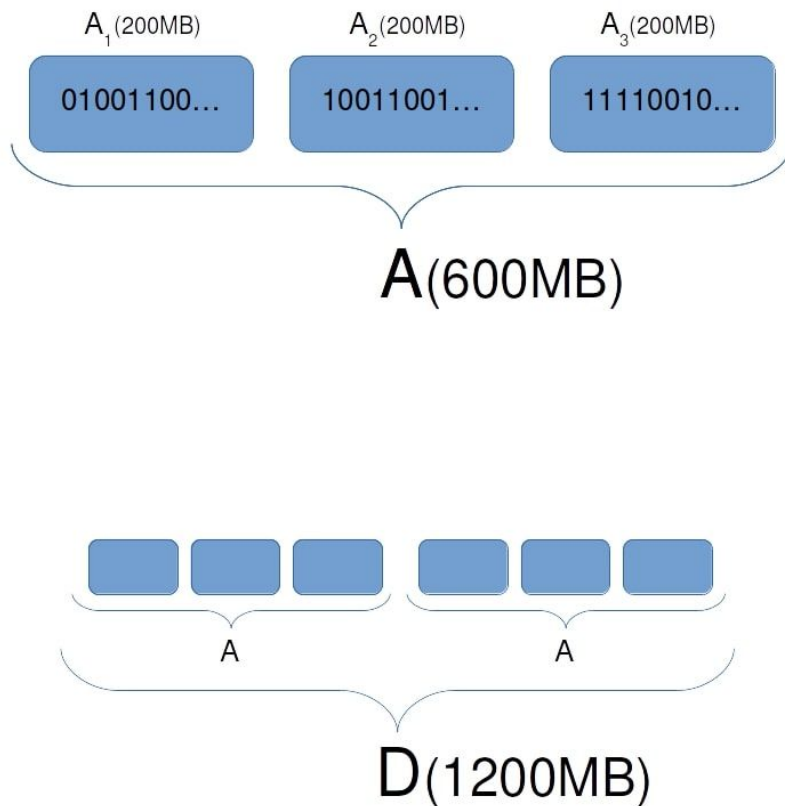
Figure 2. Constructing D

Now, A is a real pattern in D: it is repeated twice, verbatim, in the dataset. This is borne out by RP: first, A carries a great deal of information about D; and second, D is not projectible by any program computing information of a pattern that is shorter than A.

To see that the first condition is met (that A carries information about D) we simply show that $K(D|A) << K(D)$, which, as per the definition of mutual information, entails that $I(D:A)>>0$: D is made up of a repeated 600MB random (hence, incompressible) string, so $K(D)$ must be larger (but not at all much larger) than 600MB. Now, on the other hand, if we are to write a program that outputs D, and we can use A as an extra input (as per the definition of conditional Kolmogorov complexity, see above) the program can be as simple as concatenating two copies of A. Listing 1 is a function that does exactly that.[11]

---

[11]   In what follows we will give our example algorithms in pseudocode—i.e., a dialect that does not correspond to any concrete programming language, but can be readily translated to many of them and is tailored to maximize readability for humans. It may have occurred to some readers that the choice of coding scheme or programming language used to describe an object can condition the minimum achievable length for describing it and thus, seemingly, that different languages will introduce different Ks for the same object. This is correct, but the apparent relativity thus introduced does not affect the

```
reconstruct_D(A):
    return concatenate(A, A)
```

Listing 1

This program has a size of 54 bytes, approximately.[12] This is, then, an upper bound for K(D|A), and it is much smaller than K(D) ~ 600MB. A, as expected, is highly informative about D: I(D:A) = K(D) − K(D|A) ~ 600MB.

The second condition for A to be a real pattern is that D not be projectible by any program shorter than A—i.e., shorter than 600MB—and, as we have already seen, it is not: A is a random, and therefore incompressible, component of D. It is impossible for any program shorter than A to produce a copy of A and, *a fortiori*, impossible for any program shorter than A to produce a copy of D.

Now, it is also intuitively compelling that long substrings in A are real patterns in the D dataset. Suppose, for example, that D-researchers haven't yet found out about the full pattern A, but have discovered that A1 is repeated twice in D. Uncovering this fact about D would be very valuable in making sense of its structure—plausibly, the situation in genomics is analogous to this toy example in the relevant respects. Again here, RP agrees with this intuitive assessment: A1 is a real pattern in the RP sense. The program that reconstructs D given A1 as an extra input has to concatenate it with A2 and A3, concatenate the resulting string, A, with itself, then output it. A pseudocode function that does this is given in Listing 2.
[13]

```
reconstruct_D_from_A1(A1):
    A2 ← "10011001..." # The whole 200MB A2 string would be here
    A3 ← "11110010..." # And again for A3
    A ← concatenate(A1, A2, A3) # concatenates all three
substrings...
    return concatenate(A, A) # ... and everything follows as
before
```
        Listing 2

This program will be just over 400MB long, the definitions of substrings A2 and A3, each

200MB long, taking by far the biggest chunk of this space. 400MB is therefore a good estimate for K(D|A1), about two thirds of K(D). Again here, 400MB is not enough to reconstruct D, which needs at least the 600MB of A, so A1 comes out as a real pattern in D according to RP.

The problem comes with the Q pattern. This, recall, is just A1 with an extra 200MB of noise, so it's very clearly redundant in favor of A1 for the purposes of describing D. Yet, it too passes muster as a real pattern according to RP. First, it carries as much information about D as A1 does: the program that reconstructs D from Q will simply keep the first 200MB of Q, that is, A1, and then proceed as before:

```
reconstruct_D_from_Q(Q):
    A1 ← first_half(Q) # A1 is the first half of Q
    A2 ← "10011001..." # The whole 200MB A2 string would be here
    A3 ← "11110010..." # And again for A3
    A ← concatenate(A1, A2, A3) # ... and everything follows as
                                     # before
    return concatenate(A, A)
        Listing 3
```

This program has a very similar length to the one in Listing 2. That is, K(D|A1) ~ K(D|Q). And again here, while Q is double the size of A1, it is, at 400MB, still much smaller than K(D)—and, as we have explained, no string shorter than K(D) will be able to reproduce (project) D. So Q meets the RP definition of a real pattern.

In general, any string that both carries information about D and is smaller than K(D) will count as a real pattern according to RP. This, as Q shows, is an unwelcome result given the goal expressed in the non-redundancy principle: *it is trivial to multiply the redundant patterns that pass the RP test by taking a real pattern and adding to it any amount of noise such that the total length is less than the Kolmogorov complexity of the original dataset.*

6.   A Solution

How should we fix this? In the model we have developed in the previous section, the redundant pattern Q is longer than the non-redundant one A1. A quick—but obviously wrong—fix could perhaps be to rule that between any two strings, say L and S, that carry information about a target dataset, D, the longer, L, should be discarded in favor of the

shorter, S. This would discard Q in favor of A1 as desired. The problem, of course, is that the "quick fix" doesn't comment on which *particular chunks* of information the two contending strings carry about D. For example, it could be that L is identical to A1 and S to the first half of A2. In that case it would make no sense to deem L superfluous simply because it is longer than S, because both L and S make complementary contributions to our understanding of D.

What we need, if we are to decide whether one putative real pattern is redundant in favor of another, is a way of capturing when two patterns carry, not just the same *amount* of information, but the same *specific items* of information about an object. The general question of how to express this desideratum is an open problem in information theory, explored in the *partial information decomposition* framework (Griffith, Chong, James, Ellison, & Crutchfield 2014; Williams & Beer 2010) and, as far as we are aware, it has been less discussed in algorithmic information theory. Fortunately, conditional Kolmogorov complexity allows us to sneak up on the relevant notion of pattern redundancy without having to solve the general problem of partial information decomposition. We first define D-dispensability:

> **D-dispensability**: A pattern Q that carries information about a dataset D is D-dispensable iff there is another pattern P such that:
> i) $K(D|P,Q) = K(D|P)$
> ii) $K(P) < K(Q)$

Clause (i) captures the idea that the information Q carries about D is *part* of the information P carries about D. This is not a claim about information quantities, but about the actual items of information carried by these strings: if having Q as an extra input does not allow us to shave even a few bits off the length of the program that calculates D using P as input, it means that all Q has to say about D, P says it too.

Notice that if we only had clause (i), it would be possible for Q to be D-dispensable in favor of P, and P be D-dispensable in favor of Q at the same time. In keeping with the non-redundancy principle discussed above, clause (ii) deems dispensable whichever of the contending patterns is more complex.

We will say that a pattern is *strictly D-indispensable* iff it carries information about D and it is not D-dispensable. Strict D-indispensability is, we suggest, the notion that Ladyman and Ross were after in their definition of real pattern. A better definition is, thus, as simple as follows:

Real Pattern: A pattern P is real iff there is at least one dataset D such that P is strictly D-indispensable.

We offer this definition as a drop-in replacement for RP, only simpler, and without the defects presented in the previous section.

7. Concluding Remarks

We have shown that the theoretical desideratum captured in the non-redundancy principle—*include in your ontology all and only those patterns that are required for a full reconstruction of the target dataset.*—is not successfully enforced in RP. The interplay between information carrying and projection in this definition entails the reality of any string that is informative about D while being less complex than D.

We have proposed to define real patterns directly in terms of their contribution to reducing the conditional Kolmogorov complexity of a dataset. This captures the main insights in RP, while being simpler, and offering a better non-redundancy criterion. A further question[14] is whether it is sensible to tie reality to non-redundancy, as L&R and we are doing. This decision is outside the scope of this paper: we have provided clear criteria for deeming a string a *pattern* in a dataset (this is tied to compressibility), and also for deeming it a *real pattern* (this is tied to compressibility and non-redundancy). Ascertaining which one, if any, of the two notions should take priority in the metaphysics of science is matter for another work.

Our definition, as L&R's, relies on the existence of an object (a dataset in our definition, a "pattern", in theirs) which putative real patterns are more or less able to explain. It is probably possible to construe artificial datasets that would make any desired pattern real, according to any definition in the Dennettian tradition. Take any random string P. Now, we just create a "dataset" D made of two concatenated copies of P and *hey presto!* D makes P come out real. The lesson is that definitions of real patterns along Dennettian lines, such as L&R's or ours, are only interesting if the datasets appealed to are *bona fide*, in the sense of coming from the world---from actual empirical research. A more explicit characterization of what should count as a *bona fide* dataset is a central question for any broadly Dennettian account of patterns, but one that should be discussed elsewhere.

---

[14] Put to us by an anonymous referee.

References

Andersen, H. K. (2017). Pattens, Information, and Causation. *The Journal of Philosophy*, *114*(11), 592–622.

Barrett, T., Wilhite, S. E., Ledoux, P., Evangelista, C., Kim, I. F., Tomashevsky, M., … Soboleva, A. (2013). NCBI GEO: archive for functional genomics data sets—update. *Nucleic Acids Research*, *41*(D1), D991–D995. https://doi.org/10.1093/nar/gks1193

Bennett, C. H. (1995). Logical depth and physical complexity. *The Universal Turing Machine A Half-Century Survey*, 207–235.

Bird, A. P. (1986). CpG-rich islands and the function of DNA methylation. *Nature*, *321*(6067), 209. https://doi.org/10.1038/321209a0

Bogen, J., & Woodward, J. (1988). Saving the phenomena. *The Philosophical Review*, *97*(3), 303–352.

Chaitin, G. J. (1966). On the length of programs for computing finite binary sequences. *Journal of the ACM (JACM)*, *13*(4), 547–569.

Collier, J. (2001). Dealing with the Unexpected. In *Partial Proceedings of CASYS 2000: Fourth International Conference on Computing Anticipatory Systems, International Journal of Computing Anticipatory Systems*, 10:21–30.

Collier, J., and C. A. Hooker. (1999). Complexly Organised Dynamical Systems. *Open Systems & Information Dynamics* 6 (3): 241–302.

Cover, T. and Thomas, J. (2006). *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, New York, NY, USA.

Dennett, D. C. (1991). Real patterns. *The Journal of Philosophy*, *88*(1), 27–51.

Feigelson, E. D. and Babu, G. J. (2012), Big data in astronomy. Significance, 9: 22-25. doi:10.1111/j.1740-9713.2012.00587.x

Frøkjær-Jensen, C., Jain, N., Hansen, L., Davis, M. W., Li, Y., Zhao, D., … Fire, A. Z. (2016). An Abundant Class of Non-coding DNA Can Prevent Stochastic Gene Silencing in the C. elegans Germline. *Cell*, 166(2), 343–357. https://doi.org/10.1016/j.cell.2016.05.072

Gács, P., Tromp, J,.T, and Vitányi, P. (2001). Algorithmic Statistics. *IEEE Transactions on Information Theory* 47 (6): 2443–2463.

Griffith, V., Chong, E. K., James, R. G., Ellison, C. J., & Crutchfield, J. P. (2014). Intersection information based on common randomness. *Entropy*, *16*(4), 1985–2000.

Grünwald, P. (2007) *The Minimum Description Length Principle*, Cambridge: The MIT Press

Grünwald, P., Vitanyi, P (2008) Algorithmic information theory. In Eds: Adriaans, P, van Benthem, J. *Handbook of the Philosophy of Science, Philosophy of Information*, North-Holland

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

Ladyman, J., and Ross, D. (2007). *Every thing must go: Metaphysics naturalized*. Oxford University Press

Larsen, F., Gundersen, G., Lopez, R., & Prydz, H. (1992). CpG islands as gene markers in the human genome. *Genomics*, *13*(4), 1095–1107. https://doi.org/10.1016/0888-7543(92)90024-M

Li, M. and Vitányi, P. (2008). *An Introduction to Kolmogorov Complexity and Its Applications. Texts in Computer Science*. Vol. 9. Springer, New York.

McAllister, J. W. (2003a). Algorithmic randomness in empirical data. *Studies in History and Philosophy of Science Part A*, *34*(3), 633–646.

Martínez, M. (2015). Informationally-connected property clusters, and polymorphism. *Biology and Philosophy* 30 (1):99-117.

Martínez, M. (2017). Synergic Kinds. *Synthese https://doi.org/10.1007/s11229-017-1480-2*

McAllister, J. W. (2003b). Effective complexity as a measure of information content. *Philosophy of Science*, *70*(2), 302–307.

McAllister, J. W. (2011). What do patterns in empirical data tell us about the structure of the world? *Synthese*, *182*(1), 73–87.

Petersen, S. (2013). Toward an Algorithmic Metaphysics. In D. Dowe (Ed.), *Algorithmic Probability and Friends: Bayesian Prediction and Artificial Intelligence* (pp. 306–317). Springer.

Petersen, S. (2018) Composition as pattern. *Philosophical Studies. 176: 1119. https://doi.org/10.1007/s11098-018-1050-6*

Rappaport, Theodore S. (1996). *Wireless communications: principles and practice*. New Jersey: prentice hall PTR

Rissanen, J. (1998). *Stochastic complexity in statistical inquiry* (Vol. 15). World scientific.

Ross, D. (2000). Rainforest realism: A Dennettian theory of existence, in Ross, D., D. Thomson, A. Brook, eds., *Dennett's Philosophy: A Comprehensive Assessment*, 147–168. The MIT Press

Solomonoff, R. J. (1964a). A formal theory of inductive inference. Part I. *Information and Control*, *7*(1), 1–22.

Solomonoff, R. J. (1964b). A formal theory of inductive inference. Part II. *Information and Control*, *7*(2), 224–254.

Sporns, O., Tononi, G., & Kötter, R. (2005). The Human Connectome: A Structural Description of the Human Brain. *PLOS Computational Biology*, *1*(4), e42. https://doi.org/10.1371/journal.pcbi.0010042

Vereshchagin, N., and Vitányi, P. (2006). On Algorithmic Rate-Distortion Function. In *Information Theory, 2006 IEEE International Symposium On*, 798–802. IEEE.

Vereshchagin, N., and Vitanyi. P. (2010). Rate Distortion and Denoising of Individual Data Using Kolmogorov Complexity. *IEEE Transactions on Information Theory* 56 (7): 3438–54.

https://doi.org/10.1109/TIT.2010.2048491.

Vereshchagin, N., and Shen, A. (2017). Algorithmic Statistics: Forty Years Later. In *Computability and Complexity: Essays Dedicated to Rodney G. Downey on the Occasion of His 60th Birthday*, edited by Adam Day, Michael Fellows, Noam Greenberg, Bakhadyr Khoussainov, Alexander Melnikov, and Frances Rosamond, 669–737. Lecture Notes in Computer Science. Cham: Springer International Publishing. https://doi.org/10.1007/978-3-319-50062-1_41.

Williams, P. L., & Beer, R. D. (2010). Nonnegative decomposition of multivariate information. *ArXiv Preprint ArXiv:1004.2515*.