Algorithmic statistics revisited

Nikolay Vereshchagin and Alexander Shen

Abstract The mission of statistics is to provide adequate statistical hypotheses (models) for observed data. But what is an "adequate" model? To answer this question, one needs to use the notions of algorithmic information theory. It turns out that for every data string x one can naturally define "stochasticity profile", a curve that represents a trade-off between complexity of a model and its adequacy. This curve has four different equivalent definitions in terms of (1) randomness deficiency, (2) minimal description length, (3) position in the lists of simple strings and (4) Kolmogorov complexity with decompression time bounded by busy beaver function. We present a survey of the corresponding definitions and results relating them to each other.

1 What is algorithmic statistics?

The laws of celestial mechanics allow the astronomers to predict the observed motion of planets in the sky with very high precision. This was a great achievement of modern science—but could we expect to find equally precise models for all other observations? Probably not. Thousands of gamblers spent all theirs lives and their fortunes trying to discover the laws of the roulette (coin tossing, other games of chance) in the same sense—but failed. Modern science abandoned these attempts. It says modestly that all we can say about the coin tossing is the statistical hypothesis (model): *all trials are independent and* (*for a fair coin*) *both head and tail have probability* $1/2$. The task of mathematical statistics therefore is to find an appropriate model for experimental data. But what is "appropriate" in this context?

Nikolay Vereshchagin
Moscow State University and Yandex, e-mail: nikolay.vereshchagin@gmail.com

Alexander Shen
LIRMM UM2 Montpellier, on leave from IITP RAS (Moscow), e-mail: alexander.shen@lirmm.fr

To simplify the discussion, let us assume that experimental data are presented as a bit string (say, a sequence of zeros and ones corresponding to heads and tails in the coin tossing experiment). We also assume that a model is presented as a probability distribution on some finite set of binary strings. For example, a fair coin hypothesis for N coin tossings is a set of all strings of length N where all elements have the same probability 2^{-N}. Restricting ourselves to the simplest case when a hypothesis is some set A of strings with uniform distribution on it, we repeat our question:

> Assume that a bit string x (data) and a set A containing x (a model) are given; when do we consider A as a good "explanation" for x?

Some examples show that this question cannot be answered in the framework of classical mathematical statistics. Consider a sequence x of 100 bits (the following example is derived from the random tables [20]):

```
01111 10001 11110 10010 00001 00011 00001 10010 00010 11101
10111 11110 10000 11100 00111 00000 01111 01100 11011 01011
```

Probably you would agree that the statistical hypothesis of a fair coin (the set $A = \mathbb{B}^{100}$ of all 100-bit sequences) looks as an adequate explanation for this sequence. On the other hand, you probably will not accept the set A as a good explanation for the sequence y:

```
00000 00000 00000 00000 00000 00000 00000 00000 00000 00000
00000 00000 00000 00000 00000 00000 00000 00000 00000 00000
```

but will suggest a much better explanation $B = \{y\}$ (the coin that always gives heads). On the other hand, set $C = \{x\}$ does not look like a reasonable explanation for x. How can we justify this intuition?

One could say that A is not an acceptable statistical hypothesis for y since the probability of y according to A is negligible (2^{-100}). However, the probability of x for this hypothesis is the same, so *why is A acceptable for x then?* And if B looks like an acceptable explanation for y, *why C does not look as an acceptable explanation for x*?

The classical statistics, where x and y are just two equiprobable elements of A, cannot answer these questions. Informally, the difference is that x looks like a "random" element of A while y is "very special". To capture these difference, we need to use the basic notion of algorithmic information theory, Kolmogorov complexity,[1] and say that x has high complexity (cannot be described by a program that is much shorter than x itself) while y has small complexity (one can write a short program that prints a long sequence of zeros). This answers our first question and explains why A could be a good model for x but not for y.

Another question we asked: why B is an acceptable explanation for y while C is not an acceptable explanation for x? Here we need to look at the complexity of the model itself: C has high complexity (because x is complex) while B is simple (because y is simple).

[1] We assume that the reader is familiar with basic notions of algorithmic information theory and use them freely. For a short introduction see [23]; more information can be found in [15].

Now let us consider different approaches to measuring the "quality" of statistical models; they include several parameters and a trade-off between them arises. In this way for every data string x we get a curve that reflects this trade-off. There are different ways to introduce this curve, but they are all equivalent with $O(\log n)$ precision for n-bit strings. The goal of this paper is to describe these approaches and equivalence results.

2 (α,β)-stochastic objects

Let us start with the approach that most closely follows the scheme described above. Let x be a string and let A be a finite set of strings that contains x. The "quality" of A as a model (explanation) for x is measured by two parameters:

- the Kolmogorov complexity $C(A)$ of A;
- the randomness deficiency $d(x|A)$ of x in A.

The second parameter measures how "non-typical" is x in A (small values mean that x looks like a typical element of A) and is defined as

$$d(x|A) = \log \#A - C(x|A).$$

Here log stands for binary logarithm, $\#A$ is the cardinality of A and $C(u|v)$ is the conditional complexity of u given v. Using A as the condition, we assume that A is presented as a finite list of strings (say, in lexicographical ordering). The motivation for this definition: for all $x \in A$ we have $C(x|A) \le \log \#A + O(1)$, since every $x \in A$ is determined by its ordinal number in A; for most $x \in A$ the complexity $C(x|A)$ is close to $\log \#A$ since the number of strings whose complexity is much less than $\log \#A$, is negligible compared to $\#A$. So the deficiency is large for strings that are much simpler than most elements of A.[2]

According to this approach, a good explanation A for x should make both parameters small: A should be simple and x should be typical in A. It may happen that these two goals cannot be achieved simultaneously, and a trade-off arises. Following Kolmogorov, we say that x is (α,β)-*stochastic* if there exists A containing x such that $C(A) \le \alpha$ and $d(x|A) \le \beta$. In this way we get an upward closed set

$$S(x) = \{\langle \alpha,\beta \rangle \mid x \text{ is } (\alpha,\beta)\text{-stochastic}\}$$

If x is a string of length n, the set A of all n-bit strings can be used as a description; it gives us the pair $(O(\log n), n - C(x) + O(\log n))$ in $S(x)$. Indeed, we can describe

[2] There is an alternative definition of $d(x|A)$. Consider a function t of two arguments x and A, defined when $x \in A$, and having integer values. We say that t is *lower semicomputable* if there is an algorithm that (given x and A) generates lower bounds for $t(x,A)$ that converge to the true value of $t(x,A)$ in the limit. We say that t is a *probability-bounded test* if for every A and every positive integer k the fraction of $x \in A$ such that $t(x,A) > k$ is at most $1/k$. Now $d(x|A)$ can be defined as the logarithm of the maximal (up to $O(1)$-factor) lower semicomputable probability-bounded test.

A using $O(\log n)$ bits and the deficiency is $n - C(x|A) = n - C(x|n) = n - C(x) + O(\log n)$. On the other hand, there is a set $A \ni x$ of complexity $C(x) + O(1)$ and deficiency $O(1)$ (namely, $A = \{x\}$). So the boundary of the set $S(x)$ starts below the point $(0, n - C(x))$ and decreases to $(C(x), 0)$ for arbitrary n-bit string x, if we consider $S(x)$ with $O(\log n)$ precision.[3]

The boundary line of $S(x)$ can be called a *stochasticity profile* of x. As we will see, the same curve appears in several other situations.

3 Minimum description length principle

Another way to measure the "quality" of a model starts from the following observation: if x is an element of a finite set A, then x can be described by providing two pieces of information:

- the description of A;
- the ordinal number of x in A (with respect to some ordering fixed in advance).

This gives us the inequality

$$C(x) \leq C(A) + \log \#A$$

that is true with precision $O(\log n)$ for strings x of length at most n.[4]

The quality of the hypothesis A is then measured by the difference

$$\delta(x,A) = C(A) + \log \#A - C(x)$$

between the sides of this inequality. We may call it "optimality deficiency" of A, since it shows how much do we lose in the length of the description if we consider two-part description based on A instead of the best possible one. For a given string x we can then consider the set $O(x)$ of pairs $\langle \alpha, \beta \rangle$ such that x has a model of complexity at most α and optimality deficiency at most β.

Theorem 1. *For every string x of length at most n the sets $S(x)$ and $O(x)$ coincide with $O(\log n)$-precision: each of them is contained in the $O(\log n)$-neighborhood of the other one.*

Speaking about neighborhoods, we assume some standard distance on $\mathbb{R}^2$ (the exact choice does not matter, since we measure the distance up to a constant factor).

[3] As it is usual in algorithmic information theory, we consider the complexities up to $O(\log n)$ precision if we deal with strings of length at most n. Two subsets $S, T \subset \mathbb{Z}^2$ are the same for us if S is contained in the $O(\log n)$-neighborhood of T and vice verse.

[4] The additional term $O(\log C(A))$ should appear in the right hand side, since we need to specify where the description of A ends and the ordinal number of x starts, so the length of the description ($C(A)$) should be specified in advance using some self-delimiting encoding. One may assume that $C(A) \leq n$, otherwise the inequality is trivial, so this additional term is $O(\log n)$.

Let us note now that in one direction the inclusion is straightforward. A simple computation shows that the randomness deficiency is always less than the optimality deficiency *of the same model* (and the difference between them equals $C(A\,|\,x)$, where A is this model).

The opposite direction is more complicated: a model with small randomness deficiency may have large optimality deficiency. This may happens when $C(A\,|\,x)$ is large.[5] However, in this case we can find another model and decrease the optimality deficiency as needed: *for every string x and every model A for x* (a finite set A that contains x) *there exists another model A' for x such that* $\log\#(A') = \log\#A$ *and* $C(A') \le C(A) - C(A\,|\,x) + O(\log n)$, where n is the length of x. This result looks surprising at first, but note that if $C(A\,|\,x)$ is large, then there are many sets A' that are models of the same quality (otherwise A can be reconstructed from x by exhaustive search). These sets can be used to find A' with required properties.

The definition of the set $O(x)$ goes back to Kolmogorov [10]; however, he used a slightly different definition: instead of $O(x)$ he considered the function

$$h_x(\alpha) = \min_A\{\log\#A : x \in A,\ C(A) \le \alpha\},$$

now called *Kolmogorov structure function*. Both $O(x)$ and h_x are determined by the set of all pairs $(C(A), \log\#A)$ for finite sets A containing x, though in a slightly different way (since the inequality $\delta(x,A) \le \beta$ in the definition of $O(x)$ combines $C(A)$ and $\log\#A$). One can show, however, that the following statement is true with $O(\log n)$-precision for each n-bit string x: *the pair* (α,β) *is in* $O(x)$ *if and only if* $h_x(\alpha) \le \beta + C(x) - \alpha$. So the graph of h_x is just the boundary of $O(x)$ in different coordinates.

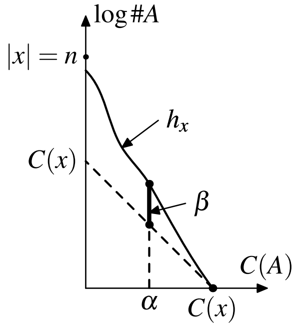

Fig. 1 The pair (α,β) lies on the boundary of $O(x)$ since the point $(\alpha, C(x) - \alpha + \beta)$ lies on the graph of h_x.

[5] Let x and y be independent random strings of length n, so the pair (x,y) has complexity close to $2n$. Assume that x starts with 0 and y starts with 1. Let A be the set of strings that start with 0, plus the string y. Then A, considered as a model for x, has large optimality deficiency but small randomness deficiency. To decrease the optimality deficiency, we may remove y from A.

4 Lists of simple strings

We have seen two approaches that describe the same trade-off between the complexity of a model and its quality: for every x there is some curve (defined up to $O(\log n)$-precision) that shows how good can be a model with bounded complexity. Both approaches gave the same curve with logarithmic precision; in this section we give one more description of the same curve.

Let m be some integer. Consider the list of strings of complexity at most m. It can be generated by a simple algorithm: just try in parallel all programs of length at most m and enumerate all their outputs (without repetitions). This algorithm is simple (of complexity $O(\log m)$) since we only need to know m.

There may be several simple algorithms that enumerate all strings of complexity at most m, and they can generate them in different orders. For example, two algorithms may start by listing all the strings of length $m-O(1)$ (they all have complexity at most m), but one does this in the alphabetical order and the other uses the reverse alphabetical order. So the string $00\ldots 00$ is the first in one list and has number $2^{m-O(1)}$ in the other. But the distance *from the end of the list* is much more invariant:

Theorem 2. *Consider two programs of complexity $O(\log m)$ that both enumerate all strings of complexity at most m. Let x be one of these strings. If there is at least 2^k strings after x in the first list, then there is at least $2^{k-O(\log m)}$ strings after x in the second list.*

In this theorem we consider two algorithms that enumerate the same strings in different orderings. However, the Kolmogorov complexity function depends on the choice of the optimal decompressor (though at most by $O(1)$ additive term), so one could ask what happens if we enumerate the strings of bounded complexity for two different versions of the complexity function. A similar result (with similar proof) says that the change of an optimal decompressor used to define Kolmogorov complexity can be compensated by $O(\log m)$-change in the threshold m.

Now for every m fix an algorithm of complexity at most $O(\log)m$ that enumerates all strings of complexity at most m. Consider a binary string x; it appears in these lists for all $m \geq C(x)$. Consider the logarithm of the number of strings that follow x in the m-th list. We get a function that is defined for all $m \geq C(x)$ with $O(\log m)$ precision. The following result shows that this function describes the stochasticity profile of x in different coordinates.

Theorem 3. *Let x be a string of length at most n.*

(**a**) *Assume that x appears in the list of strings of complexity at most m and there are at least 2^k strings after x in the list. Then the pair $((m-k)+O(\log n), m-C(x))$ belongs to the set $O(x)$.*

(**b**) *Assume that the pair $(m-k, m-C(x))$ belongs to $O(x)$. Then x appears in the list of strings of complexity at most $m+O(\log n)$ and there are at least $2^{k-O(\log n)}$ strings after it.*

By Theorem 1 the same statement holds for the set $S(x)$ in place of $O(x)$.

Ignoring the logarithmic correction and taking into account the relation between $O(x)$ and h_x, one can illustrate the statement of Theorem 3 by Figure 2.

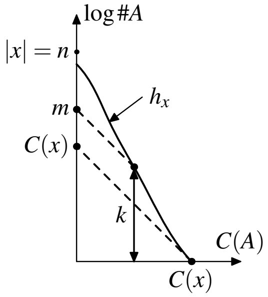

Fig. 2 To find how many strings appear after x in the list of all strings of complexity at most m, we draw a line starting at $(0,m)$ with slope -1 and intersect it with the graph of h_x; if the second coordinate of the intersection point is k, there are about 2^k strings after x in this list.

5 Time-bounded complexity and busy beavers

There is one more way to get the stochasticity profile curve. Let us bound the computation time (number of steps) in the definition of Kolmogorov complexity and define $C^t(x)$ as the minimal length of a program that produces x in at most t steps. Evidently, $C^t(x)$ decreases as t increases, and ultimately reaches $C(x)$.[6] However, the convergence speed may be quite different for different x of the same complexity. It is possible that for some x the programs of minimal length produce x rather fast, while other x can be compressible only if we allow very long computations. Informally, the strings of the first type have some simple internal structure that allows us to encode them efficiently with a fast decoding algorithm, while the strings of the second type have "deep" internal structure that is visible only if the observer has a lot of computational power.

We use the so-called "busy beaver numbers" as landmarks for measuring the computation time. Let $BB(n)$ be the maximal running time of all programs of length at most n (we use the programming language that defines Kolmogorov complexity,

[6] One may ask which computational model is used to measure the computation time, and complain that the notion of time-bounded complexity may depend on the choice of an optimal programming language (decompressor) and its interpreter. Indeed this is the case, but we will use very rough measure of computation time based on busy beaver function, and the difference between computational models does not matter. The reader may assume that we fix some optimal programming language, and some interpreter (say, a Turing machine) for this language, and count the steps performed by this interpreter.

and some fixed interpreter for it).[7] One can show that numbers $BB(n)$ have equivalent definition in terms of Kolmogorov complexity: *$BB(n)$ is the maximal integer that has complexity at most n.* (More precisely, if $B(n)$ is the maximal integer that has complexity at most n, then $B(n-c) \leq BB(n) \leq B(n+c)$ for some c and all n, and we ignore $O(1)$-changes in the argument of the busy beaver function.)

Now for every x we may consider the decreasing function $i \mapsto C^{BB(i)}(x) - C(x)$ (it decreases fast for "shallow" x and slowly for "deep" x; note that it becomes close to 0 when $i = C(x)$, since then every program of length at most $C(x)$ terminate in $BB(C(x))$ steps.). The graph of this function is (with logarithmic precision) just a stochasticity profile, i.e., the set of points above the graph coincides with $O(x)$ up to a $O(\log n)$ error term:

Theorem 4. *Let x be a string of length n.*

(**a**) *If a pair (α, β) is in $O(x)$, then*

$$C^{BB(\alpha+O(\log n))}(x) \leq C(x) + \beta + O(\log n).$$

(**b**) *If $C^{BB(\alpha)}(x) \leq C(x) + \beta$, then the pair $(\alpha + O(\log n), \beta + O(\log n))$ is in $O(x)$.*

By Theorem 1 the same statement holds for the set $S(x)$ in place of $O(x)$.

6 How the stochasticity profile can look like?

We have seen four different definitions that lead to the same (with logarithmic precision) notion of stochasticity profile. We see now that finite objects (strings) not only could have different complexities, but also the strings with the same complexity can be classified according to their stochasticity profiles.

However, we do not know yet that this classification is non-trivial: what if all strings of given complexity have the same stochasticity profile? The following result answers this question by showing that every simple decreasing function appears as complexity profile of some string.

Theorem 5. *Assume that some integers n and $k \leq n$ are given, and h is a non-increasing function mapping $\{0, 1, \ldots, k\}$ to $\{0, 1, \ldots, n-k\}$. Then there exists a string x of length $n + O(\log n) + O(C(h))$ and complexity $k + O(\log n) + O(C(h))$ for which the set $O(x)$ (and hence the set $S(x)$) coincides with the upper-graph of h (the set $\{\langle i, j\rangle \mid j \geq h(i)$ or $i \geq k\}$) with $O(\log n + C(h))$ accuracy.*

Note that the error term depends on the complexity of h. If we consider simple functions h, this term is absorbed by our standard error term $O(\log n)$. In particular,

[7] Usually n-th busy beaver number is defined as the maximal running time or a maximal number of non-empty cells that can appear after Turing machine with at most n states terminates starting on the empty tape. This gives a different number; we modify the definition so it does not depend on the peculiarities of encoding information by transition tables of Turing machines.

this happens in two extreme cases: for the function $h \equiv 0$ and the function h that is equal to $n-k$ everywhere. In the first case it is easy to find such a "shallow" x: just take an incompressible string of length k and add $n-k$ trailing zeros to get a n-bit string. For the second case we do not know a better example than the one obtained from the proof of Theorem 5.

Let us say informally that a string x of length n is "stochastic" if its stochasticity profile $S(x)$ is close to the maximal possible set (achieved by the first example) with logarithmic precision, i.e., x is $(O(\log n), O(\log n))$-stochastic. We know now that non-stochastic objects exist in the mathematical sense; a philosopher could ask whether they appear in the "real life". Is it possible that some experiment gives us data that do not have any adequate statistical model? This question is quite philosophical since having an object and a model we cannot say for sure whether the model is adequate in terms of algorithmic statistics. For example, the current belief is that the coin tossing data are described adequately by a fair coin model. Still it is possible that future scientists will discover some regularities in the very same data, thus making this model unsuitable.

We discuss the properties of stochastic objects in the next section. For now let us note only that this notion remains essentially the same if we consider probability distributions (and not finite sets) as models. Let us explain what does it mean.

Consider a probability distribution P on a finite set of strings with rational values. It is a constructive object, so we can define the complexity of P using some computable encoding. The conditional complexity $C(\cdot\,|\,P)$ can be defined in the same way. Let us modify the definition of stochasticity and say that a string x is "(α,β)-p-stochastic" if there exists a distribution P of the described type such that

- $C(P)$ is at most α;
- $d(x\,|\,P)$, defined as $-\log P(x) - C(x\,|\,P)$, does not exceed β.

This is indeed a generalization: if P is a uniform distribution, then the complexity of P is (up to $O(1)$) the complexity of its support A, the value of $-\log P(x)$ is $\log \#A$, and using P and A as conditions gives the same complexity up to $O(1)$. On the other hand, this generalization leads only to a logarithmic change in the parameters:

Theorem 6. *If some string x of length n is (α,β)-p-stochastic, then the stting x is also $(\alpha + O(\log n), \beta + O(\log n))$-stochastic.*

Since all our statements are made with $O(\log n)$-precision, we may identify stochasticity with p-stochasticity (as we do in the sequel).

7 Canonical models

Let Ω_m denote the number of strings of complexity at most m. Consider its binary representation, i.e., the sum

$$\Omega_m = 2^{s_1} + 2^{s_2} + \ldots + 2^{s_t}, \text{ where } s_1 > s_2 > \ldots > s_t.$$

According to this decomposition, we may split the list itself into groups: first 2^{s_1} elements, next 2^{s_2} elements, etc.[8] If x is a string of complexity at most m, it belongs to some group, and this group can be considered as a model for x.

We may consider different values of m (starting from $C(x)$). In this way we get different models of this type for the same x. Let us denote by $B_{m,s}$ the group of size 2^s that appears in the m-th list. Note that $B_{m,s}$ is defined only for s that correspond to ones in the binary representation of Ω_m. The models $B_{m,s}$ are called *canonical* models in the sequel. The parameters of $B_{m,s}$ are easy to compute: the size is 2^s by definition, and the complexity is $m-s+O(\log m)$.

Theorem 7. (**a**) *Every canonical model for a string x lies on the boundary of $O(x)$ (i.e., its parameters cannot be improved more than by $O(\log n)$ where n is the length of x).*

(**b**) *For every point in $O(x)$ there exists a canonical model that has the same or better parameters (with $O(\log n)$ precision).*

The second part of this theorem says that for every model A for x we can find a canonical model $B_{m,s}$ that has the same (or smaller) optimality deficiency, and $C(B_{m,s}) \le C(A)$ with logarithmic precision. In fact, the second part of this statement can be strengthened: not only $C(B_{m,s}) \le C(A)$, but also $C(B_{m,s}\,|\,A) = O(\log n)$.

This result shows that (in a sense) we may restrict ourselves to canonical models. This raises the question: what are these models? What information they contain? The answer is a bit confusing: the information in models $B_{m,s}$ depends on $m-s$ only and is the same as the information in Ω_{m-s}, the number of strings of complexity at most $m-s$:

Theorem 8. *For all models $B_{m,s}$ both conditional complexities $C(B_{m,s}\,|\,\Omega_{m-s})$ and $C(\Omega_{m-s}\,|\,B_{m,s})$ are $O(\log m)$.*

One could note also that the information in Ω_k is a part of the information in Ω_l for $l \ge k$ (i.e., $C(\Omega_k\,|\,\Omega_l) = O(\log l)$).[9]

Now it seems that finding a good model for x does not provide any specific information about x: all we get (if we succeed) is the information about the number of terminating programs of bounded length, which that had nothing to do with x and is the same for all x.

It is not clear how this philosophical collision between our goals and our achievements can be resolved. One of the approaches is to consider *total conditional complexity*. This approach still leaves many questions open, but let us shortly describe it nevertheless.

We have said that "string a and b contain essentially the same information" if both $C(a\,|\,b)$ and $C(b\,|\,a)$ are small. This, however, does not guarantee that the properties of a and b are the same. For example, if x^* is the shortest program for some highly non-stochastic string x, the string x^* itself is perfectly stochastic.

[8] We assume that an algorithm is fixed that, given m, enumerates all strings of complexity at most m in some order.

[9] In fact, Ω_k contains the same information (up to $O(\log k)$ conditional complexity in both directions) as first k bits of Chaitin's Ω-number (a lower semicomputable random real), so we use the same letter Ω to denote it.

To avoid this problem, we can consider total condition complexity $CT(a\,|\,b)$ defined as the minimal length of a *total* program p such that $p(b)=a$. Here p is called total if $p(b')$ halts for all b', not only for b.[10] This total conditional complexity can be much bigger than the standard conditional complexity $C(a\,|\,b)$. It has the following property: if both $CT(a\,|\,b)$ and $CT(b\,|\,a)$ are negligible, there exists a computable permutation of low complexity that maps b to a, and therefore the sets $O(a)$ and $O(b)$ are close to each other. (See [17] for more details.)

Using this notion, we may consider a set A as a "strong" model if it is close to the boundary of $O(x)$ and at the same time the *total* complexity $CT(A\,|\,x)$ is small. The second condition is far from trivial: one can prove that for some strings x such strong models do not exist at all (except for the trivial model $\{x\}$ and the models of very small complexity) [27]. But if strong models exists, they have some nice properties: for example, the stochasticity profile of every strong sufficient statistic for x is close to the profile of the string x itself [26]. (A model is called a sufficient statistic for x if the optimality deficiency is small, i.e., the sum of its complexity and log-cardinality is close to $C(x)$.) The class of all sufficient statistics for x does not have this property (for some x).

Returning to the stochasticity profile, let us mention one more non-existence result. Imagine that we want to find a place when the set $O(x)$ touches the horizontal coordinate line. To formulate a specific task, consider for a given string of length n two numbers. The first, α_1, is the maximal value of α such that $(\alpha, 0.1n)$ does not belong to $O(x)$; the second, α_2, is the minimal value of α such that $(\alpha, 10\log n)$ belongs to $O(x)$. (Of course, the constant 10 is chosen just to avoid additional quantifiers, any sufficiently large constant would work.) Imagine that we want, given x and $C(x)$, to find some point in the interval (α_1, α_2), or even in a slightly bigger one (say, adding the margin of size $0.1n$ in both directions). One can prove that *there is no algorithm that fulfills this task* [24].

8 Stochastic objects

The philosophical questions about non-stochastic objects in the "real world" motivate several mathematical questions. Where do they come from? can we obtain a non-stochastic object by applying some (simple) algorithmic transformation to a stochastic one? Can non-stochastic objects appear (with non-negligible probability) in a (simple) random process? What are the special properties of non-stochastic objects?

Here are several results answering these questions.

Theorem 9. *Let f be a computable total function. If string x of length n is (α,β)-stochastic, then $f(x)$ is $(\alpha + C(f) + O(\log n), \beta + C(f) + O(\log n))$-stochastic.*

Here $C(f)$ is the complexity of the program that computes f.

[10] As usual, we assume that the programming language is optimal, i.e., gives $O(1)$-minimal value of the complexity compared to other languages.

An important example: let f the projection function that maps every pair $\langle x,y\rangle$ (its encoding) to x. Then we have $C(f)=O(1)$, so we conclude that each component of an (α,β)-stochastic pair is $(\alpha+O(\log n),\beta+O(\log n))$-stochastic.

A philosopher would interpret Theorem 9 as follows: *a non-stochastic object cannot appear in a simple total algorithmic process* (unless the input was already non-stochastic). Note that the condition of totality is crucial here: for every x, stochastic or not, we may consider its shortest program p. It is incompressible (and therefore stochastic), and x is obtained from p by a simple program (decompressor).

If a non-stochastic object cannot be obtained by a (simple total) algorithmic transformation from a stochastic one, can it be obtained (with non-negligible probability) in a (simple computable) random process? If P is a simple distribution on a finite set of strings with rational values, then P can be used as a statistical model, so only objects x with high randomness deficiency $d(x\,|\,P)$ can be non-stochastic, and the set of all x that have $d(x\,|\,P)$ greater than some d has negligible P-probability (an almost direct consequence of the deficiency definition).

So for computable probabilistic distributions the answer is negative for trivial reasons. In fact, much stronger (and surprising) statement is true. Consider a probabilistic machine M without input that, being started, produces some string and terminates, or does not terminate at all (and produces nothing). Such a machine determines a *semimeasure* on the set of strings (we do not call it measure since the sum of probabilities of all strings may be less than 1 if the machine hangs with positive probability). The following theorem says that a (simple) machine of this type produces non-stochastic objects with negligible probability.

Theorem 10. *There exists some constant c such that the probability of the event*

“M produces a string of length at most n that is not
$(d+C(M)+c\log n, c\log n)$*-stochastic”*

is bounded by 2^{-d} *for every machine M of described type and for arbitrary integers n and d.*

The following results partially explain why this happens. Recall that algorithmic information theory defines *mutual information* in two strings x and y as $C(x)+C(y)-C(x,y)$; with $O(\log n)$ precision (for strings of length at most n) this expression coincides with $C(x)-C(x\,|\,y)$ and $C(y)-C(y\,|\,x)$. Recall that by Ω_n we denote the number of strings of complexity at most n.

Theorem 11. *There exists a constant c such that for every n, for every string x of length at most n and for every threshold d the following holds: if a string x of length n is not* $(d+c\log n, c\log n)$*-stochastic, then*

$$I(x:\Omega_n)\geq d-c\log n.$$

This theorem says that all non-stochastic objects have a lot of information about a specific object, the string Ω_n. This explain why they have small probability to appear in a (simple) randomized process, as the following result shows. It guarantees that for every fixed string w the probability to get (in a simple random process) some object that contains significant information about w, is negligible.

Theorem 12. *There exists a constant c such that for every n, for every probabilistic machine M, for every string w of length at most n and for every threshold d the probability of the event*

"M outputs a string x of length at most n such that $I(x : w) > C(M) + d + c\log n$"

is at most 2^{-d}.

The last result of this section shows that stochastic objects are "representative" if we are interested only in the complexity of strings and their combinations: for every tuple of strings one can find a stochastic tuple that is indistinguishable from the first one by complexities of its components.

Theorem 13. *For every k there exists a constant c such that for every n and for every k-tuple $\langle x_1, \ldots, x_k\rangle$ of strings of length at most n, there exist another k-tuple $\langle y_1, \ldots, y_k\rangle$ that is $(c\log n, c\log n)$-stochastic and for every $I \subset \{1, 2, \ldots, n\}$ the difference between $C(x_I)$ and $C(y_I)$ is at most $c\log n$.*

Here x_I is a tuple made of strings x_i with $i \in I$; the same for y_I.

This result implies, for example, that every linear inequality for complexities that is true for stochastic tuples, is true for arbitrary ones.

However, there are some results that are known for stochastic tuples but still are not proven for arbitrary ones. See [18] for details.

9 Restricted classes: Hamming distance and balls as descriptions

Up to now we considered arbitrary sets as statistical models. However, sometimes we have some external information that suggests a specific class of models (and it remains to choose the parameters that define some model in this class). For example, if the data string is a message sent through a noisy channel that can change some bits, we consider Hamming balls as models, and the parameters are the center of this ball (the original message) and its radius (the maximal number of changed bits).

So let us consider some family $\mathscr{B}$ of finite sets. To get a reasonably theory, we need to assume some properties of this family:

- The family $\mathscr{B}$ is computably enumerable: there exists an algorithm that enumerates all elements of $\mathscr{B}$ (finite sets are here considered as finite objects, encoded as lists of their elements).
- For each n the set of all n-bit strings belongs to $\mathscr{B}$.
- There exists a polynomial p such that the following property holds: for every $B \in \mathscr{B}$, for every positive integer n and for every $c < \#B$ the set of all n-bit strings in B can be covered by $p(n)\#B/c$ sets from $\mathscr{B}$ and each of the covering sets has cardinality at most c.

Here #B stands for the cardinality of B. Counting argument shows that in the last condition we need at least #B/c covering sets; the condition says that polynomial overhead is enough here.

One can show (using simple probabilistic arguments) that the family of all Hamming balls (for all string lengths, centers and radii) has all three properties. This family is a main motivating example for our considerations.

Now we can define the notion of a $\mathscr{B}$-(α,β)-stochastic object: a string x is $\mathscr{B}$-(α,β)-stochastic if there exists a set $B\in\mathscr{B}$ containing x such that $C(B)\le\alpha$ and $d(x|B)\le\beta$. (The original notion of (α,β)-stochasticity corresponds to the case when $\mathscr{B}$ contains all finite sets.) For every x we get a set $S_{\mathscr{B}}(x)$ of pairs (α,β) for which x is $\mathscr{B}$-(α,β)-stochastic. We can also define the set $O_{\mathscr{B}}(x)$ using optimality deficiency instead of randomness deficiency. The $\mathscr{B}$-version of Theorem 1 is still true (though the proof needs a much more ingenious construction):

Theorem 14. *Let $\mathscr{B}$ be the family of finite sets that has the properties listed above. Then the for every string x of length at most n the sets $S_{\mathscr{B}}(x)$ and $O_{\mathscr{B}}(x)$ coincide up to a $O(\log n)$ error term.*

The proof is more difficult (compared to the proof of Theorem 1) since we now need to consider sets in $\mathscr{B}$ instead of arbitrary finite sets. So we cannot construct the required model for a given string x ourselves and have to select it among the given sets that cover x. This can be done by a game-theoretic argument.

It is interesting to note that a similar argument can be used to obtain the following result about stochastic finite set (Epstein–Levin theorem):

Theorem 15. *If a finite set X is (α,β)-stochastic and the total probability*

$$\sum_{x\in X}2^{-K(x)}$$

of its elements exceeds 2^{-k}, then X contains some element x such that

$$K(x)\le k+K(k)+\log K(k)+\alpha+O(\log\beta)+O(1).$$

Here $K(u)$ stands for the prefix complexity of u (see, e.g., [15] for the definition). To understand the meaning of this theorem, let us recall one of the fundamental results of the algorithmic information theory: the (prefix) complexity of a string x equals the binary logarithm of its a priori probability. If we consider a set X of strings instead of one string x, we can consider the a priori probability of X (expressing how difficult is to get some element of x in a random process) and the minimal complexity of elements of X (saying how difficult is to specify an individual element in X). The fundamental result mentioned above says that for singletons these two measures are closely related; for arbitrary finite sets it is no more the case, but Theorem 15 guarantees that for the case for *stochastic* finite sets.

Returning to our main topic, let us note that for Hamming balls the boundary curve of $O_{\mathscr{B}}(x)$ has a natural interpretation. To cover x of length n with a ball B with center y having cardinality 2^β and complexity at most α means (with logarithmic precision) to find a string y of complexity at most α in the r-neighborhood of x, where r is chosen is such a way that balls of radius r have about 2^β elements. So this boundary curve represents a trade-off between the complexity of y and its distance to x.

Again one can ask what kind of boundary curves may appear. As in Theorem 5, we can get essentially arbitrary non-increasing function. However, here precision is worse: $O(\log n)$ term is now replaced by $O(\sqrt{n \log n})$.

Theorem 16. *Assume that some integers n and $k \leq n$ are given, and h is a non-increasing function mapping $\{0,1,\ldots,k\}$ to $\{0,1,\ldots,n-k\}$. Then there exists a string x of length $n + O(\sqrt{n \log n}) + O(C(h))$ and complexity $k + O(\sqrt{n \log n}) + O(C(h))$ for which the set $O(x)$ coincides with the upper-graph of h (the set $\{\langle i,j\rangle \mid j \geq h(i)$ or $i \geq k\}$) with $O(\sqrt{n \log n} + C(h))$-precision.*

Unlike the general case where non-stochastic objects (for which the curve is far from zero) exists but are difficult to describe, for the case of Hamming balls one can give more explicit examples. Consider some explicit error correction code that has distance d. Then every string that differs in at most $d/2$ positions from some codeword x, has almost the same complexity as x (since x can be reconstructed from it by error correction). So the balls of radius less than $d/2$ containing some codeword have almost the same complexity as the codeword itself (and the balls of zero radius containing it).

Let x be a typical codeword of this binary code (its complexity is close to the logarithm of the number of codewords). For values of α slightly less than $C(x)$ we need a large β (at least the logarithm of the cardinality of a ball of radius $d/2$) to make such a codeword (α,β)-stochastic.

10 Historical remarks

The notion of (α,β)-stochasticity was mentioned by Kolmogorov in his talks at the seminar he initiated at the Moscow State University in early 1980s (see [22]). The equivalence between this notion and the optimality deficiency (Theorem 1) was discovered in [24].

The connections between the existence of adequate models and the position in the list of strings of bounded complexity was discovered by Gács, Tromp and Vitányi in [5], though this paper considered only the position of x in the list of strings of complexity at most $C(x)$. Theorems 2 and 3 appeared in [24]. The paper [5] considered also the canonical models (called “nearly sufficient statistics” in this paper) for the case $m = C(x)$. In the general case the canonical models were considered in [24] (section V, *Realizing the structure function*), where Theorems 7 and 8 were proven.

The minimal description length principle goes back to Rissanen [16]; as he wrote in this paper, “If we work with a fixed family of models, $\langle\ldots\rangle$ the cost of the complexity of a model may be taken as the number of bits it takes to describe its parameters. Clearly now, when adding new parameters to the model, we must balance their own cost against the reduction they permit in the ideal code length, $-\log P(x\,|\,\theta)$, and we get the desired effect in a most natural manner. If we denote the total number of bits required to encode the parameters θ by $L(\theta)$, the we can write the total code length as $L(x,\theta) = -\log P(x\,|\,\theta) + L(\theta)$, which we seek to minimize over θ". The set denoted by $O(x)$ in our survey was considered in 1974 by Kolmogorov (see [10]); later it appeared in the literature also under the names of "sophistication" and "snooping curves".

The notion of sophistication was introduced by Koppel in [12]. Let β be a natural number; *β-sophistication* of a string x is the minimal length of a total program p such that there is a string y with $p(y) = x$ and $|p| + |y| \le C(x) + \beta$. In out terms p defines a model that consists of all $p(y)$ for all strings y of a given length. It is not hard to see that with logarithm precision we get the same notion: the β-sophistication of x is at most α if and only if the pair (α, β) is in the set $O(x)$.

The notion of snooping curve $L_x(\alpha)$ of x was introduced by V'yugin in [31]. In this paper he considered strategies that read a bit sequence from left to right and for each next bit provide a prediction (a rational-valued probability distribution on the set $\{0,1\}$ of possible outcomes). After the next bit appears, the *loss* is computed depending on the prediction and actual outcome. The goal of the predictor is to minimize the total loss, i.e., the sum of losses at all n stages (for a n-bit sequence). Vyugin considered different loss functions, and for one of them, called *logarithmic loss function*, we get a notion equivalent to $O(x)$. For a logarithmic loss function, we account for loss $-\log p$ if the predicted probability of the actual outcome was p. It is easy to see that for a given x the following statement is true (with logarithmic precision): there exists a strategy of complexity at most α with loss at most l if and only if $l \ge h_x(\alpha)$. (Indeed, prediction strategies are just bit-by-bit representation of probability distributions on the set of n-bit strings, in terms of conditional probabilities.)

Theorem 4 (Section 5) is due to Bauwens [2]. The idea to consider the difference between time bounded complexity of x and the unbounded one goes back to Chaitin [6]. Later the subject was studied by Bennett who introduced the notion of logical depth: the *depth of x at significance level β* is the minimal time t such that $C^t(x) \le C(x) + \beta$. The string is called (β,t)-deep if its depth at significance level β is larger than t. A closely related notion of computational depth was introduced in [1]: the *computational depth of x with time bound t* is $C^t(x) - C(x)$. Obviously, computational depth of x with time bound t is more than β if and only if x is (β,t)-deep. Theorem 4 relates both notions of depth to the stochasticity profile (with logarithmic precision): a string is $(\beta, B(\alpha))$-deep if and only if the pair (α, β) is outside the set $O(x)$.

Theorem 5 was proved in [24]. Long before this paper (in 1987) V'yugin established that set $S(x)$ can assume all possible shapes (within the obvious constraints) but only for $\alpha = o(|x|)$. Also, according to Levin [14]: "Kolmogorov told me about $h_x(\alpha)$ and asked how it could behave. I proved that $h_x(\alpha) + \alpha + O(\log \alpha)$ is monotone but otherwise arbitrary within $\pm O(p \log \alpha)$ accuracy where p is the number of "jumps" of the arbitrary function imitated; it stabilizes on $C(x)$ when α exceeds

$I(\chi : x)$ [the information in the characteristic sequence χ of the "halting problem" about x]. The expression for accuracy was reworded by Kolmogorov to $O(\sqrt{\alpha \log \alpha})$ [*square root accuracy*]; I gave it in the above, less elegant, but equivalent, terms. He gave a talk about these results at a meeting of Moscow Mathematical Society [11]." This claim of Levin implies Theorem 11 that was published in [24].

Theorem 6 (mentioned in [22]) is easy and Theorem 9 easily follows from Theorem 5.

The existence of non-(α, β)-stochastic strings (for small α, β) was mentioned in [22]. Then V'yugin [29] and Muchnik [19] showed that that their a priori measure is about $2^{-\alpha}$, a direct corollary of which is our Theorem 10.

Theorems 11 and 12 are essentially due to Levin (see [14] and [13]).

Theorem 13 is easy to prove using A. Romashchenko's "typization" trick (see [9, 21]).

Theorems 14 and 16 appeared in [25]; Theorem 15 appeared in [8].

Acknowledgements Authors thank all their colleagues with whom they discussed algorithmic statistics, especially Bruno Bauwens, Laurent Bienvenu, Sam Epstein, Peter Gács, Leonid Levin, Paul Vitányi, Vladimir V'yugin, and all the members of the Kolmogorov seminar (Moscow) and ESCAPE team (Marseille, Montpellier).

The preparation of this survey was supported in part by the EMC ANR-09-BLAN-0164 and RFBR 12-01-00864 grant.

References

1. L. Antunes, L. Fortnow, D. van Melkebeek, and N. Vinodchandran. Computational depth: Concept and applications. *Theoretical Computer Science*, **354**:3, 391–404
2. B. Bauwens, *Computability in statistical hypotheses testing, and characterizations of independence and directed influences in time series using Kolmogorov complexity*, Ph.D. thesis, University of Gent, May 2010.
3. C.H. Bennett, Logical Depth and Physical Complexity, in *The Universal Turing Machine: a Half-Century Survey* R. Herken, ed., Oxford University Press, 1988, 227–257.
4. P. Gács, Attending [10], e-mail to Paul Vitanyi, January 24, 2002.
5. P. Gács, J. Tromp, P.M.B. Vitányi. Algorithmic statistics, *IEEE Transactions on Information Theory*, **47**:6 (2001), 2443–2463.
6. G.J Chaitin. Algorithmic information theory, IBM J. Research Developments, **21** (1977), 350–359.
7. T.M. Cover, Attending [10], Email to Paul Vitanyi, January 24, 2002.
8. S. Epstein, L.A. Levin, *Sets Have Simple Members*, arxiv:1107.1458v8 (2011–2014)
9. D. Hammer, A. Romashchenko, A. Shen, N. Vereshchagin, Inequalities for Shannon entropy and Kolmogorov complexity, *Journal of Computer and System Sciences*, **60** (2000), 442–464.
10. A.N. Kolmogorov, Talk at the Information Theory Symposium in Tallinn, Estonia, 1974, according to [4, 7].
11. A.N. Kolmogorov, Complexity of algorithms and objective definition of randomness, *Uspekhi Mat. Nauk*, **29**:4 (1974), 155 (Abstract of a talk at the meeting of the Moscow Mathematical Society, April 16, 1974, in Russian.)
12. M. Koppel. Structure, in *The Universal Turing Machine: A Half-Century Survey*. R. Herken, ed., Oxford University Press, 1988, 435–452.

13. L.A. Levin, Randomness conservation inequalities; information and independence in mathematical theories, *Information and Control*, **61**:1 (1984), 15–37.
14. L.A. Levin. Emails to Paul Vitányi (February 7,11, and 20, 2002).
15. M. Li, P.M.B. Vitányi, *An introduction to Kolmogorov complexity and its applications*, 3rd ed., Springer, 2008, 792 p. ISBN 978-0–387-49820-1.
16. J. Rissanen, A Universal Prior for Integers and Estimation by Minimum Description Length, *The Annals of Statistics*, **11**:2 (1983), 416–431.
17. An. Muchnik, I. Mezhirov, A. Shen, N. Vereshchagin, *Game interpretation of Kolmorogov complexity*, `arxiv.org/abs/1003.4712`.
18. An.A. Muchnik, A.E. Romashchenko, Stability of properties of Kolmogorov complexity under relativization, *Problems of Information Transmission*, **46**:1 (2010), 38–61. (Preliminary version: Random oracle does not help extract the mutual information, MFCS 2008, LNCS, **5162**, 527–538.)
19. An.A. Muchnik , A.L. Semenov, V.A. Uspensky, Mathematical metaphysics of randomness, *Theoretical Computer Science*, **207**:2 (November 1998), 263–317.
20. RAND Corporation. *A million random digits with 100,000 normal deviates.* Glencoe, Ill. : Free Press, 1955.
21. A. Romashchenko, A. Shen, N. Vereshchagin, Combinatorial interpretation of Kolmogorov complexity, *Theoretical Computer Science*, **271**:1–2 (2002), 111–123.
22. A. Shen, The concept of (α,β)-stochasticity in the Kolmogorov sense, and its properties. *Soviet Mathematics Doklady*, **271**:1 (1983), 295–299.
23. A. Shen, Algorithmic information theory and Kolmogorov complexity, Lecture notes , Uppsala University TR2000-034, `www.it.su.se/research/publications/reports/2000-034`. See also p. 000–000 of this volume.
24. N. Vereshchagin, P. Vitanyi. Kolmogorov's Structure Functions with an Application to the Foundations of Model Selection, *IEEE Transactions on Information Theory*, **50**:12 (2004), 3265–3290. Preliminary version: *Proc.* 47*th IEEE Symp. Found. Comput. Sci.*, 2002, 751–760.
25. N.K. Vereshchagin, P.M.B. Vitányi. Rate Distortion and Denoising of Individual Data Using Kolmogorov Complexity, *IEEE Transactions on Information Theory*, **56**:7 (2010), 3438–3454.
26. N. Vereshchagin, Algorithmic Minimal Sufficient Statistic Revisited, in *Mathematical Theory and Computational Practice, 5th Conference on Computability in Europe*, CiE 2009, Heidelberg, Germany, July 19–24, 2009. Proceedings, LNCS 5635, 478–487.
27. N. Vereshchagin. On Algorithmic Strong Sufficient Statistics. In: 9th Conference on Computability in Europe, CiE 2013, Milan, Italy, July 1-5, 2013. Proceedings, LNCS 7921, P. 424–433.
28. V.V. V'yugin, On the defect of randomness of a finite object with respect to measures with given complexity bounds, *SIAM Theory Probab. Appl.*, **32**:3 (1987), 508–512.
29. V.V. V'yugin, Nonstochastic objects, *Problems of Information Transmission*, **21**:2 (1985), 77–83.
30. V.V. V'yugin, Algorithmic Complexity and Stochastic Properties of Finite Binary Sequence, *Computer Journal*, **42**:4 (1999), 294–317, `dx.doi.org/10.1093/comjnl/42.4.294`.
31. V.V. V'yugin. Does snooping help? *Theoretical Computer Science*, **276**:1 (2002), 407–415.