Discussion on Kolmogorov Complexity and Statistical Analysis

ALEXANDER SHEN

125284, Moskva, Begovaja ul., 17-14, Russia
Email: ashen@mccme.ru

The question why and how probability theory can be applied to the real-world phenomena has been discussed for several centuries. When the algorithmic information theory was created, it became possible to discuss these problems in a more specific way. In particular, Li and Vitányi [6], Rissanen [3], Wallace and Dowe [7] have discussed the connection between Kolmogorov (algorithmic) complexity and minimum description length (minimum message length) principle. In this note we try to point out a few simple observations that (we believe) are worth keeping in mind while discussing these topics.

1. TWO-PART DESCRIPTION OF AN OBJECT

Let p be a program that enumerates some finite set A (generates its elements; the program does not need to terminate explicitly). Assume that all elements of A are binary strings of length at most n. Then, for all $x \in A$,

$$K(x) \leq K(p) + \log_2 |A| + O(\log n). \qquad (1)$$

Here $K(p)$ denotes the Kolmogorov complexity of program p and $|A|$ stands for the cardinality of A. There are different versions of Kolmogorov complexity (the original one, prefix complexity and others, see [1]), but they differ by at most $O(\log n)$ and the difference is not important here.

The inequality (1) could be explained as follows: any object $x \in A$ has a two-part description. The first part is (a description of a) program p. The second part is the number of x in the enumeration of A (the element that appears first has number 1, the next element has number 2, etc.). The first part requires $K(p)$ bits. The second part requires at most $\log_2 |A|$ bits. (Additional $O(\log n)$ bits are needed to form a pair; we omit the details.)

We are interested in 'efficient' two-part descriptions for which the inequality (1) is close to equality. For any string x there are many efficient descriptions. Here are two 'extreme' examples:

(a) The set A consists of x only: $A = \{x\}$; the program p that enumerates A just prints x and its complexity is $K(x)$. Since $|A| = 1$, the second term $\log_2 |A|$ vanishes.

(b) Let $k = K(x)$. Let A be the set of all strings that have complexity at most k and length at most n. The complexity of the program that generates this set is $O(\log n)$ (to generate it one must know only n and k; since $k \leq n$, only $O(\log n)$ bits are needed). On the other hand, $|A| = O(2^k)$, so $\log_2 |A| \leq k + O(1)$.

It is easy to mix these two examples in an arbitrary proportion. Let s be any integer between 0 and $K(x)$. Divide the set A from example (b) into 2^s disjoint subsets. Element number m goes into subset $A_{m \bmod 2^s}$. Then each subset A_u has complexity $s + O(\log n)$ (we need s bits to specify u) and at most 2^{k-s} elements. Therefore, the terms in the right-hand side of (1) are $s + O(\log n)$ and $k - s$ and this two-part description is also efficient (in the sense explained above).

2. TWO-PART DESCRIPTIONS REVISITED

However, the situation changes completely if we require the program p that enumerates A to stop after the last element of A appears. In other words, now we consider the complexity of the list of elements of A (let us call it the 'listing complexity' of A and denote it by $K(A)$) instead of the complexity of a program that enumerates A ('enumerating complexity' of A).

The example (a) is still valid, since the complexity of the one-element list $\langle x \rangle$ is $K(x)$. However, the example (b) disappears, since listing complexity of the set $A = \{x\colon K(x) \leq k\}$ is close to k, not to $\log k$ (as it was for enumerating complexity). Indeed, assume that we know the list of all elements of A. Then we can construct the smallest element z not in A. By definition, z has complexity more than k. On the other hand, its complexity cannot exceed significantly the listing complexity of A.

For some strings x a two-part description of type (b) still exists. For example, if x is a random (incompressible) string of length n ($K(x) = n$), then we can consider the set of all strings of length n as A. Its listing complexity is $O(\log n)$ and $\log_2 A = n$.

For other strings a two-part description of type (b), i.e. an efficient two-part description with small $K(A)$, is impossible (see [2] for an exact statement and proof). Kolmogorov suggested calling such objects 'non-stochastic'.

The trick used to shift the balance between two parts of the description (move some bits from the second part to the first one) still works. Indeed, for any set A and for any integer s we can divide A into 2^s sets of size about $|A|/2^s$. Each of them can be described by $K(A) + s$ bits. So the length of the first part of the description increases by s while the length of the second part ($\log_2 |A|$) decreases by s.

3. EFFICIENT DESCRIPTIONS AND RANDOM ELEMENTS

Conditional complexity $K(x \mid y)$ is defined as the minimal length of a program that transforms y to x. One can prove that

$$K(\langle x, y\rangle) \approx K(x) + K(y \mid x) \qquad (2)$$

for any strings x and y. Here $\approx$ means that both sides differ at most by $O(\log(K(x) + K(y)))$ and $\langle x, y\rangle$ stands for a pair composed from x and y (written using some natural encoding of pairs).

The two-part description describes not only $x \in A$, but the whole pair $\langle A, x\rangle$, so

$$K(x) \leq K(\langle A, x\rangle) \leq K(|A|) + \log_2 |A| + O(\log n). \quad (3)$$

So A provides an efficient two-part description if both inequalities in (3) are close to equalities, i.e. if

$$K(x) \approx K(\langle A, x\rangle) \quad \text{and} \quad K(\langle A, x\rangle) \approx K(A) + \log_2 |A|.$$

Using equation (2), these two conditions could be rewritten as

$$K(A \mid x) \approx 0 \quad \text{and} \quad K(x \mid A) \approx \log_2 |A|.$$

The first condition means that A is simple when x is known; the second means that x is a 'random' element of A. (The difference $\log_2 |A| - K(x \mid A)$ is often considered as 'randomness deficiency' of x in A measuring how far x is from being a random element of A.)

4. STATISTICAL INTERPRETATION

The idea of Kolmogorov's minimum sufficient statistic is explained in [3] as follows:

> First, a 'summarizing' property of data may be formalized as a subset A where the data belongs along with other sequences sharing this property. Hence, the property A need not specify the sequence completely. We may now think of programs consisting of two parts, where the first part describes optimally the set A with the number of bits given by the Kolmogorov complexity $K(A)$ and the second part merely describes x^n as A with about $\log|A|$ bits, $|A|$ denoting the number of elements in A. The sequence x^n then gets described in $K(A) + \log|A|$ bits. We may now ask for a set $\hat{A}$ for which $K(\hat{A})$ is minimal subject to the constraint that for an increasing length sequence x^n, $K(\hat{A}) + \log|\hat{A}|$ agrees with the Kolmogorov complexity $K(x^n)$ to within a constant not depending on n. The set $\hat{A}$, or its defining program, may be called Kolmogorov's 'minimal sufficient statistic' for the description of x^n. The bits describing $\hat{A}$ are then the 'interesting' bits in the program (code) while the rest, about $\log|\hat{A}|$ in number, are non-informative noise bits.

This description does not specify, however, how A should be presented: as a (possibly non-terminating) program that generates elements of A or as a list of all elements ('$\approx$' a program that generates all the elements of A and then terminates). As we have seen, the choice of representation is very important. Most probably Kolmogorov had in mind the second possibility when he asked about the existence of non-stochastic sequences.

From this point of view, the goal of a scientist observing some binary string x as a result of an experiment is to provide an 'explanation' for x. Such an explanation is a set A that is as simple as possible ($K(A)$ is minimal) but still is good for x (i.e. x belongs to A and has a small 'randomness deficiency' $\log_2 |A| - K(x)$). Non-stochastic objects are objects that do not allow the scientist to explain them. (One may ask whether such objects appear in the real world.)

5. SETS AND DISTRIBUTIONS

The scheme presented above is oversimplified: normally the 'statistical explanation' of data string x is not only some set A (containing x) but also some probability distribution P on A. It is convenient to consider P as a distribution on the set of all finite binary strings (all strings x such that $P(x) > 0$ form the set A).

The inequality (1) is replaced now by the following one:

$$K(x) \leq K(P) + (-\log_2 P(x)) + O(\log n). \qquad (4)$$

Here P is a probability distribution on binary strings, $P(x)$ is the probability assigned to string x. Let us note that if P is a uniform distribution on some set A, then $(-\log_2 P(x))$ is equal to $\log_2 |A|$ (so we return to (1)). What is $K(P)$? As before, there are two different ways to define this notion.

1. Consider a machine that has access to random bits and may print a binary string on its output. After the string is printed the machine terminates. Then each string x appears as output with some probability $P(x)$. The sum $\sum P(x)$ over all strings x does not exceed one. This sum may be less than one since the computation may never stop (with some positive probability). So any machine M of this type determines some distribution P on binary strings (with additional 'undefined' element) and the complexity of M's program is considered as $K(P)$. (Different Ms could define the same P; in this case we take the simplest one.)
2. Another approach is to consider probability distributions with finite domain and rational values. Such a probability distribution can be encoded by a binary string in a natural way and $K(P)$ is the Kolmogorov complexity of that string.

(There are several variations of these approaches. For example, if in the first approach we require that the machine terminates with probability one, we get a notion that is close to the second approach. Also we may allow computable reals in the second approach instead of rational numbers, etc.)

These two approaches are parallel to the two ways to measure the complexity of the set A explained before. For both of them the inequality (4) is valid. Any distribution P determines a two-part description of x. The first part describes P; the second part describes x when P is known. When P was a uniform distribution on some set A, each element of A was described by its number. Now we have to use a more efficient description like the Shannon–Fano encoding where the encoding length is close to $-\log p$ (here p is the probability of an object).

As before, for both approaches the balance between parts of a two-part efficient description may be shifted in one direction (complexity may be transferred from $\log_2 P(x)$ into $K(P)$), but for the second approach the way back is not always possible. If $K(P)$ is defined in the second way, there are some 'non-stochastic' strings x such that there is no two-part efficient description for x with small $K(P)$.

The 'randomness deficiency' for a string x with respect to distribution P could be defined as $-\log_2 P(x) - K(x \mid P)$; this definition is a generalization of a definition given above.

6. LIMITATIONS: TIME BOUNDS AND PSEUDO-RANDOMNESS

The definitions and statements given above may serve as a basis for some philosophical speculations. For example (as we have said) one may ask whether non-stochastic objects exist in the real world. If yes, such an object will pose an unsolvable problem for scientists who never could provide a satisfactory explanation for it, etc.

One should be very careful, however, since all the definitions above do not take into account the computational resources needed for encoding/decoding. The following example shows the problem that may arise. Consider a pseudo-random number generator (for the definition see [4]) that maps, say, a 1000-bit random seed into a 1,000,000-bit pseudo-random string p. Such a string will have small complexity (it is equal to the complexity of the seed, i.e. 1000) so p is highly non-random. (Even the time-bounded complexity of p is small since the pseudo-random number generator is computationally efficient.) An efficient two-part description of p exists: the first part is the description of the generator, the second part is random seed.

However, imagine that somebody gives us a black box. Inside this box there is a random generator that produces a 1000-bit random seed when the box is turned on and a chip that implements a pseudo-random number generator. We do not know what is inside the box, we just connect the box to a printer, turn the box on and get a 1,000,000-bit string p printed. Is there any chance that we can find out the efficient two-way description of p or discover the internal structure of the box? The current belief is that it is impossible, such a string will seem random to us forever. So another two-part description for p where A is the set of all binary strings of length 1,000,000 will be considered as an efficient one and we never will find out that it is not the case.

Therefore, complexity considerations should not be considered as something really practical, they can only give some hints for real applications and motivate our decisions. Let us show one example of the latter type.

There is a general rule saying that a statistical hypothesis P could be rejected if P assigns a small probability to a simple set T containing the data string x (see, e.g., [5] for examples). We can support this rule by the following inequality: if T is a finite set of binary strings and P is a probability distribution on binary strings, then

$$K(x \mid P) \leq -\log_2 P(x) + \log_2 P(T) + K(T) + \text{logarithmic terms}$$

(we omit the details). This inequality says that randomness deficiency in x with respect to P is at least $\log_2(1/P(T)) - K(T)$ (up to logarithmic terms) so if $P(T)$ is small and T is simple, the randomness deficiency is large and the hypothesis P does not 'explain' the data string x.

REFERENCES

[1] Uspensky, V. A. and Shen, A. (1996) Relations between varieties of Kolmogorov complexities. *Math. Systems Theory*, **29**, 271–292.

[2] Shen, A. (1983) The concept of (α, β)-stochasticity and its properties. *Sov. Math.–Dokl.*, **28**, 295–299.

[3] Rissanen, J. (1999) Hypotheses selection and testing by the MDL principle. *Comput. J.*, **42**, 260–269.

[4] Luby, M. (1996) *Pseudorandomness and Cryptographic Applications*. Princeton University Press, Princeton, NJ.

[5] Uspensky, V. A., Semenov, A. L. and Shen, A. (1990) Can an individual sequences of zeros and ones be random? *Russian Math. Surveys*, **45**, 121–189.

[6] Li, M. and Vitányi, P. (1997) *An Introduction to Kolmogorov Complexity and its Applications* (2nd edn). Springer, New York.

[7] Wallace, C. S. and Dowe, D. L. (1999) Minimum message length and Kolmogorov complexity. *Comput. J.*, **42**, 270–283.