Kolmogorov's Structure Functions and Model Selection

Nikolai Vereshchagin and Paul Vitányi

Abstract— **In 1974 Kolmogorov proposed a non-probabilistic approach to statistics, an individual combinatorial relation between the data and its model, expressed by the so-called "structure function" of the data. We show that the structure function determines all stochastic properties of the data in the sense of determining the best-fitting model at every model-complexity level. A consequence is this: minimizing the data-to-model code length (finding the ML estimator or MDL estimator), in a class of contemplated models of prescribed maximal (Kolmogorov) complexity, *always* results in a model of best fit, irrespective of whether the source producing the data is in the model class considered. In this setting, code minimization *always* separates optimal model information from the remaining accidental information, and not only with high probability. The function that maps the maximal allowed model complexity to the goodness-of-fit (expressed as minimal "randomness deficiency") of the best model cannot itself be monotonically approximated. However, the shortest one-part or two-part code above can—implicitly optimizing this elusive goodness-of-fit. We show that—within the obvious constraints—every graph is realized by the structure function of some data. We determine the (un)computability properties of the various functions contemplated and of the "algorithmic minimal sufficient statistic."**

I. Introduction

As perhaps the last mathematical innovation of an extraordinary scientific career, Kolmogorov [16], [15] proposed to found statistical theory on finite combinatorial principles independent of probabilistic assumptions. Technically, the new statistics is expressed in terms of Kolmogorov complexity, [14], the information in an individual object. The relation between the individual data and its explanation (model) is expressed by Kolmogorov's structure function. This function, its variations and its relation to model selection, have obtained some notoriety [21], [3], [26], [6], [13], [22], [27], [10], [12], [9], [4], but it has not before been comprehensively analyzed and understood. It has always been questioned why Kolmogorov chose to focus on the the mysterious function h_x below, rather than on the more evident β_x variant below. The only written record by Kolmogorov himself is the following abstract [15] (translated by L.A. Levin):

"To each constructive object corresponds a function $\Phi_x(k)$ of a natural number k—the log of minimal cardinality of x-containing sets that allow definitions of complexity at most k. [We use h_x in place of Φ_x.] If the element x itself allows a simple definition, then the function Φ drops to 1 [presumably, $0 = \log 1$ is meant] even for small k. Lacking such definition, the element is "random" in a negative sense. But it is positively "probabilistically random" only when function Φ having taken the value Φ_0 at a relatively small $k = k_0$, then changes approximately as $\Phi(k) = \Phi_0 - (k - k_0)$."

These pregnant lines will become clear on reading this paper. Our main result, with the beauty of truth, establishes the importance of the structure function: For all data, minimizing a two-part code consisting of one part model description and one part data-to-model code (essentially the two-part MDL estimator [18]), subject to a given model-complexity constraint, *in every case* (and not only with high probability) selects a model that is a best explanation (within a certain negligible tolerance) of the data within the given model-complexity constraint. The same holds for minimizing the one-part code consisting of just the data-to-model code (essentially the maximum likelihood estimator). The explanatory value of an individual model for particular data is quantified by by the randomness deficiency (II.6): minimal randomness deficiency implies that the data is maximally "random" or "typical" for the model. It turns out that the minimal randomness deficiency of a model for the data, the minimum taken over the elements of the contemplated model class, cannot be computationally monotonically approximated up to any significant precision. Thus, while we can monotonically approximate the minimal length two-part code, or the one-part code, and thus monotonically approximate *implicitly* the best fitting model, we cannot monotonically approximate the number expressing the goodness of this fit. But this should be sufficient: we want the best model rather than a number that measures its goodness.

Monotonic Approximation: In technical terms our notion of monotonic approximation means the existence of a non-halting algorithm A that given any x, α outputs a finite sequence $p_1, p_2, p_3, \ldots, p_l$ of pairwise different computer programs each of length at most $\alpha + C \log |x|$ (C is a constant) such that each program p_i prints a model S_i such that $|S_1| > |S_2| > \cdots > |S_l|$, and the last model S_l is best in the following sense. There is no program p of length at most α that prints a model S such that the randomness deficiency of x for S is $C \log |x|$ less than that of x for S_l. Note that we are not able to identify p_l given x, α, since the algorithm A is non-halting and thus we do not know

Manuscript received xxx, 2002; revised yyy 2003. Part of this work was done during N.K. Vershchagin's stay at CWI. His work was supported in part by Grant 01-01-01028 from Russian Federation Basic Research Fund and by CWI. The work of P.M.B. Vitányi was supported in part by the EU fifth framework project QAIP, IST–1999–11234, the NoE QUIPROCONE IST–1999–29064, the ESF QiT Programmme, and the EU Fourth Framework BRA NeuroCOLT II Working Group EP 27150. N.K. Vereshchagin is with the Department of Mathematical Logic and Theory of Algorithms, Faculty of Mechanics and Mathematics, Moscow State University, Leninskie Gory, Moscow, Russia 119992. Email: ver@mech.math.msu.su. P.M.B. Vitányi is with CWI, Kruislaan 413, 1098 SJ Amsterdam, The Netherlands. Email: Paul.Vitányi@cwi.nl.

which program will be output last.

Epistimology: Classical statistics investigates real-world phenomena using probabilistic methods. There is the problem of what probability means, whether it is subjective, objective, or exists at all. The total probability concentrated on potentially realizable data may be negligible, for example, in complex video and sound data. In such a case, a model selection process that is successful with high probability may nonetheless fail on the actually realized data. Kolmogorov's proposal strives for the firmer and less contentious ground of finite combinatorics and effective computation.

Reach of Results: In Kolmogorov's initial proposal, as in this work, models are finite sets of finite binary strings, and the data is one of the strings (all discrete data can be binary encoded). The restriction to finite set models is just a matter of convenience: the main results generalize to the case where the models are arbitrary computable probability density functions, [21], [1], [22], [10], and to the model class consisting of arbitrary total recursive functions, [24]. We summarize the proofs of this below. Since our results hold only within additive logarithmic precision, and the equivalences between the model classes hold up to the same precision, the results hold equally for the more general model classes.

The generality of the results are at the same time a restriction. In classical statistics one is commonly interested in model classes that are partially poorer and partially richer than the ones we consider. For example, the class of Bernoulli processes, or k-state Markov chains, is poorer than the class of computable probability density functions of moderate maximal Kolmogorov complexity α, in that the latter may contain functions that require far more complex computations than the rigid syntax of the former classes allows. Indeed, the class of computable probability density functions of even moderate complexity allows implementation of a function mimicking a universal Turing machine computation. On the other hand, even the lowly Bernoulli process can be equipped with a noncomputable real bias in $(0,1)$, and hence the generated probability density function over n trials is not a computable function. This incomparability of the here studied algorithmic model classes, and the traditionally studied statistical model classes, means that the current results cannot be directly transplanted to the traditional setting. They should be regarded as pristine truths that hold in a platonic world that can be used as guideline to develop analogues in model classes that are of more traditional concern, [19].

II. Preliminaries

Self-delimiting Code: Let $x,y,z \in \mathcal{N}$, where $\mathcal{N}$ denotes the natural numbers and we identify $\mathcal{N}$ and $\{0,1\}^*$ according to the correspondence

$$(0,\epsilon),(1,0),(2,1),(3,00),(4,01),\ldots$$

Here ϵ denotes the *empty word*. The *length* $|x|$ of x is the number of bits in the binary string x, not to be confused with the *cardinality* $|S|$ of a finite set S. For example, $|010| = 3$ and $|\epsilon| = 0$, while $|\{0,1\}^n| = 2^n$ and $|\emptyset| = 0$. The emphasis is on binary sequences only for convenience; observations in any alphabet can be so encoded in a way that is 'theory neutral'.

A binary string y is a *proper prefix* of a binary string x if we can write $x = yz$ for $z \neq \epsilon$. A set $\{x,y,\ldots\} \subseteq \{0,1\}^*$ is *prefix-free* if for any pair of distinct elements in the set neither is a proper prefix of the other. A prefix-free set is also called a *prefix code* and its elements are called *code words*. An example of a prefix code, that is useful later, encodes the source word $x = x_1x_2\ldots x_n$ by the code word

$$\overline{x} = 1^n0x.$$

This prefix-free code is called *self-delimiting* because we can determine where the code word $\bar{x}$ ends by reading it from left to right without backing up. (This desirable property holds for every prefix-free encoding of a finite set of source words, but not for every prefix-free encoding of an infinite set of source words.) Using this code we define the standard self-delimiting code for x to be $x' = \overline{|x|}x$. It is easy to check that $|\overline{x}| = 2n+1$ and $|x'| = n + 2\log n + 1$. Let $\langle \cdot \rangle$ denote a standard invertible effective one-one encoding from $\mathcal{N} \times \mathcal{N}$ to a subset of $\mathcal{N}$. For example, we can set $\langle x,y \rangle = x'y$ or $\langle x,y \rangle = \bar{x}y$. We can iterate this process to define $\langle x, \langle y,z \rangle \rangle$, and so on.

Kolmogorov Complexity: For precise definitions, notation, and results see the text [13]. Informally, the Kolmogorov complexity, or algorithmic entropy, $K(x)$ of a string x is the length (number of bits) of a shortest binary program (string) to compute x on a fixed reference universal computer (such as a particular universal Turing machine). Intuitively, $K(x)$ represents the minimal amount of information required to generate x by any effective process. The conditional Kolmogorov complexity $K(x|y)$ of x relative to y is defined similarly as the length of a shortest program to compute x, if y is furnished as an auxiliary input to the computation. For technical reasons we use a variant of complexity, so-called prefix complexity, which is associated with Turing machines for which the set of programs resulting in a halting computation is prefix free. We realize prefix complexity by considering a special type of Turing machine with a one-way input tape, a separate work tape, and a one-way output tape. Such Turing machines are called *prefix* Turing machines. If a machine T halts with output x after having scanned all of p on the input tape, but not further, then $T(p) = x$ and we call p a *program* for T. It is easy to see that $\{p : T(p) = x, x \in \{0,1\}^*\}$ is a *prefix code*. Let $T_1, T_2, \ldots$ be a standard enumeration of all prefix Turing machines with a binary input tape, for example the lexicographical length-increasing ordered syntactic prefix Turing machine descriptions, [13], and let $\phi_1, \phi_2, \ldots$ be the enumeration of corresponding functions that are computed by the respective Turing machines (T_i computes ϕ_i). These functions are the *partial recursive* functions or *computable* functions (of effectively prefix-free encoded arguments). The Kolmogorov complexity of x is the length of the shortest binary program from which x is computed.

Definition II.1: The *prefix Kolmogorov complexity* of x is

$$K(x) = \min_{p,i}\{|\bar{i}| + |p| : T_i(p) = x\}, \quad \text{(II.1)}$$

where the minimum is taken over $p \in \{0,1\}^*$ and $i \in \{1, 2, \ldots\}$. For the development of the theory we actually require the Turing machines to use *auxiliary* (also called *conditional*) information, by equipping the machine with a special read-only auxiliary tape containing this information at the outset. Then, the *conditional version* $K(x \mid y)$ of the prefix Kolmogorov complexity of x given y (as auxiliary information) is is defined similarly as before, and the unconditional version is set to $K(x) = K(x \mid \epsilon)$.

One of the main achievements of the theory of computation is that the enumeration $T_1, T_2, \ldots$ contains a machine, say $U = T_u$, that is computationally universal in that it can simulate the computation of every machine in the enumeration when provided with its index: $U(\langle y, \bar{i}p\rangle) = T_i(\langle y, p\rangle)$ for all i, p, y. We fix one such machine and designate it as the *reference universal prefix Turing machine.* Using this universal machine it is easy to show $K(x \mid y) = \min_q\{|q| : U(\langle y, q\rangle) = x\}$.

A prominent property of the prefix-freeness of $K(x)$ is that we can interpret $2^{-K(x)}$ as a probability distribution since $K(x)$ is the length of a shortest prefix-free program for x. By the fundamental Kraft's inequality, see for example [6], [13], we know that if $l_1, l_2, \ldots$ are the code-word lengths of a prefix code, then $\sum_x 2^{-l_x} \leq 1$. Hence,

$$\sum_x 2^{-K(x)} \leq 1. \quad \text{(II.2)}$$

This leads to the notion of universal distribution—a rigorous form of Occam's razor—which implicitly plays an important part in the present exposition. The functions $K(\cdot)$ and $K(\cdot \mid \cdot)$, though defined in terms of a particular machine model, are machine-independent up to an additive constant and acquire an asymptotically universal and absolute character through Church's thesis, from the ability of universal machines to simulate one another and execute any effective process. The Kolmogorov complexity of an individual object was introduced by Kolmogorov [14] as an absolute and objective quantification of the amount of information in it. The information theory of Shannon [20], on the other hand, deals with *average* information *to communicate* objects produced by a *random source.* Since the former theory is much more precise, it is surprising that analogs of theorems in information theory hold for Kolmogorov complexity, be it in somewhat weaker form. An example is the remarkable *symmetry of information* property used later (denoting $K(x, y) = K(\langle x, y\rangle)$):

$$\begin{aligned} K(x, y) &= K(x) + K(y \mid x, K(x)) + O(1) \quad \text{(II.3)}\\ &= K(y) + K(x \mid y, K(y)) + O(1). \end{aligned}$$

The meaning of the pair $x, K(x)$ in the condition of this expression is as follows: Let x^* denote the shortest program x^* for a finite string x, or, if there are more than one of these, then x^* is the first one halting in a fixed standard enumeration of all halting programs. The information in the pair $(x, K(x))$ is equal to the information in x^*: Given x^* we can compute $x = U(x^*)$ and $K(x) = |x^*|$, and given $x, K(x)$ we can run all programs of length $K(x)$ simultaneously, round-robin fashion, until the first program computing x halts—this is by definition x^*.

Precision: It is customary in this area to use "additive constant c" or equivalently "additive $O(1)$ term" to mean a constant, accounting for the length of a fixed binary program, independent from every variable or parameter in the expression in which it occurs. In this paper we use the prefix complexity variant of Kolmogorov complexity for convenience. Actually some results, especially Theorem B.1, are easier to prove for plain complexity. Most results presented here are precise up to an additive logarithmic term, which means that they are valid for plain complexity as well—prefix complexity exceeds plain complexity by at most a logarithmic additve term. Thus, our use of prefix complexity is important for "fine details" only.

Meaningful Information: The information contained in an individual finite object (like a finite binary string) is measured by its Kolmogorov complexity—the length of the shortest binary program that computes the object. Such a shortest program contains no redundancy: every bit is information; but is it meaningful information? If we flip a fair coin to obtain a finite binary string, then with overwhelming probability that string constitutes its own shortest program. However, also with overwhelming probability all the bits in the string are meaningless information, random noise. On the other hand, let an object x be a sequence of observations of heavenly bodies. Then x can be described by the binary string pd, where p is the description of the laws of gravity, and d the observational parameter setting: we can divide the information in x into meaningful information p and accidental information d. The main task for statistical inference and learning theory is to distil the meaningful information present in the data. The question arises whether it is possible to separate meaningful information from accidental information, and if so, how. The essence of the solution to this problem is revealed when we rewrite (II.1) as follows (use the universality of the fixed reference universal prefix Turing machine $U = T_u$ with $|u| = O(1)$ to obtain the last equality):

$$\begin{aligned} K(x) &= \min_{p,i}\{|\bar{i}p| : T_i(p) = x\} \quad \text{(II.4)}\\ &= \min_{p,i}\{2|i| + |p| + 1 : T_i(p) = x\},\\ &= \min_{p,i}\{K(i) + |p| : T_i(p) = x\} + O(1), \end{aligned}$$

where the minimum is taken over $p \in \{0,1\}^*$ and $i \in \{1, 2, \ldots\}$. This expression emphasizes the two-part code nature of Kolmogorov complexity. In the example

$$x = 10101010101010101010101010$$

we can encode x by a small Turing machine printing a specified number of copies of the pattern "01" which computes x from the program "13." This way, $K(x)$ is viewed as the shortest length of a two-part code for x, one part describing a Turing machine, or *model*, for the *regular* aspects of x, and the second part describing the *irregular* aspects of x in the form of a program to be interpreted by T. The regular, or "valuable," information in x is constituted by the bits in the "model" while the random or "useless" information of x constitutes the remainder.

Data and Model: To simplify matters, and because all discrete data can be binary coded, we consider only finite binary data strings x. Our model class consists of Turing machines T that enumerate a finite set, say S, such that on input $p \leq |S|$ we have $T(p) = x$ with x the pth element of T's enumeration of S, and $T(p)$ is a special *undefined* value if $p > |S|$. The "best fitting" model for x is a Turing machine T that reaches the minimum description length in (II.4). Such a machine T embodies the amount of useful information contained in x, and we have divided a shortest program x^* for x into parts $x^* = T^*p$ such T^* is a shortest self-delimiting program for T. Now suppose we consider only low complexity finite-set models, and under these constraints the shortest two-part description happens to be longer than the shortest one-part description. Does the model minimizing the two-part description still capture all (or as much as possible) meaningful information? Such considerations require study of the relation between the complexity limit on the contemplated model classes, the shortest two-part code length, and the amount of meaningful information captured.

Kolmogorov Structure Functions: We will prove that there is a close relation between functions describing three, a priori seemingly unrelated, aspects of modeling individual data by models of prescribed complexity: minimal remaining randomness, optimal fit, and length of shortest two-part code, respectively (Figure 1). We first need a definition. Denote the *complexity of the finite set* S by $K(S)$—the length (number of bits) in the shortest binary program p from which the reference universal prefix machine U computes a listing of the elements of S and then halts. That is, if $S = \{x_1, \ldots, x_n\}$, then $U(p) = \langle x_1, \langle x_2, \ldots, \langle x_{n-1}, x_n \rangle \ldots \rangle\rangle$. The shortest program p, or, if there is more than one such shortest program, then the first one that halts in a standard dovetailed running of all programs, is denoted by S^*. For every finite set $S \subseteq \{0,1\}^*$ containing x we have

$$K(x|S) \leq \log |S| + O(1). \tag{II.5}$$

Indeed, consider the selfdelimiting code of x consisting of its $\lceil \log |S| \rceil$ bit long index of x in the lexicographical ordering of S. This code is called *data-to-model code*. Its length quantifies the maximal "typicality," or "randomness," data (possibly different from x) can have with respect to this model. The lack of typicality of x with respect to S is measured by the amount by which $K(x|S)$ falls short of the length of the data-to-model code. The *randomness deficiency* of x in S is defined by

$$\delta(x|S) = \log |S| - K(x|S), \tag{II.6}$$

for $x \in S$, and ∞ otherwise. The *minimal randomness deficiency* function is

$$\beta_x(\alpha) = \min_S \{\delta(x|S) : S \ni x,\ K(S) \leq \alpha\}, \tag{II.7}$$

where we set $\min \emptyset = \infty$. The smaller $\delta(x|S)$ is, the more x can be considered as a *typical* member of S. This means that a set S for which x incurs minimal deficiency is a "best" model for x—a most likely explanation, and $\beta_x(\alpha)$ can be viewed as a *fitness function*. If the randomness deficiency is close to 0, then are no simple special properties that single it out from the majority of elements in S. This is not just terminology: If $\delta(x|S)$ is small enough, then x satisfies *all* properties of low Kolmogorov complexity that hold with high probability for the elements of S. To be precise: Consider strings of length n and let S be a subset of such strings.

(i) If P is a property satisfied by all x with $\delta(x|S) \leq \delta(n)$, then P holds with probability at least $1 - 1/2^{\delta(n)}$ for the elements of S.

(ii) Let P be any property that holds with probability at least $1 - 1/2^{\delta(n)}$ for the elements of S. Then, every such P holds simultaneously for every $x \in S$ with $\delta(x|S) \leq \delta(n) - K(P|S) - O(1)$.

Example II.2: **Lossy Compression:** The function $\beta_x(\alpha)$ is relevant to lossy compression (used, for instance, to compress images). Assume we need to compress x to α bits where $\alpha \ll K(x)$. Of course this implies some loss of information present in x. One way to select redundant information to discard is as follows: Find a set $S \ni x$ with $K(S) \leq \alpha$ and with small $\delta(x|S)$, and consider a compressed version S' of S. To reconstruct an x', a decompresser uncompresses S' to S and selects at random an element x' of S. Since with high probability the randomness deficiency of x' in S is small, x' serves the purpose of the message x as well as does x itself. Let us look at an example. To transmit a picture of "rain" through a channel with limited capacity α, one can transmit the indication that this is a picture of the rain and the particular drops may be chosen by the receiver at random. In this interpretation, $\beta_x(\alpha)$ indicates how "random" or "typical" x is with respect to the best model at complexity level α—and hence how "indistinguishable" from the original x the randomly reconstructed x' can be expected to be. ◊

Kolmogorov [16], [15] proposed the following function:

$$h_x(\alpha) = \min_S \{\log |S| : S \ni x,\ K(S) \leq \alpha\}, \tag{II.8}$$

where $S \ni x$ is a contemplated model for x, and α is a nonnegative integer value bounding the complexity of the contemplated S's. We call h_x the *Kolmogorov structure* function for data x. Clearly, this function is non-increasing and reaches $\log |\{x\}| = 0$ for $\alpha = K(x) + c_1$ where c_1 is the number of bits required to change x into $\{x\}$. The function can also be viewed as the *maximum likelihood (ML)* function, a viewpoint that is more evident for its version for probability models, Figure 2. For every $S \ni x$ we have

$$K(x) \leq K(S) + \log |S| + O(1). \tag{II.9}$$

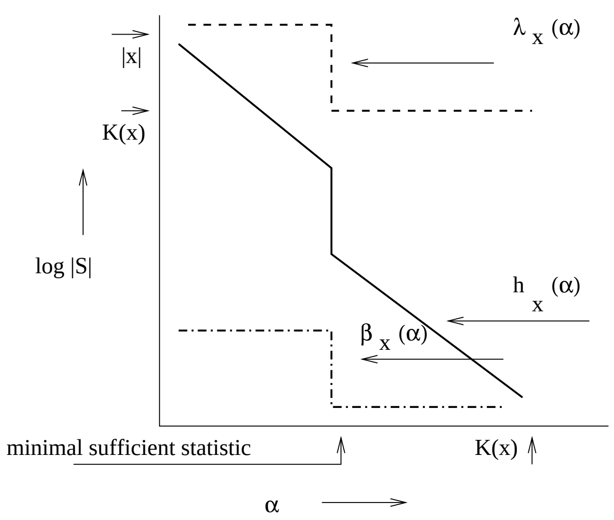

Fig. 1. Structure functions $h_x(\alpha), \beta_x(\alpha), \lambda_x(\alpha)$, and minimal sufficient statistic.

Indeed, consider the following *two-part code* for x: the first part is a shortest self-delimiting program p of S and the second part is $\lceil \log |S| \rceil$ bit long index of x in the lexicographical ordering of S. Since S determines $\log |S|$ this code is self-delimiting and we obtain (II.9) where the constant $O(1)$ is the length of the program to reconstruct x from its two-part code. We thus conclude that $K(x) \leq \alpha + h_x(\alpha) + O(1)$, that is, the function $h_x(\alpha)$ never decreases more than a fixed independent constant below the diagonal *sufficiency line* L defined by $L(\alpha)+\alpha = K(x)$, which is a lower bound on $h_x(\alpha)$ and is approached to within a constant distance by the graph of h_x for certain α's (for instance, for $\alpha = K(x) + c_1$). For these α's we have $\alpha + h_x(\alpha) = K(x) + O(1)$ and the associated model (witness for $h_x(\alpha)$) is called a *sufficient statistic*. The sufficient statistic associated with the least such α is called the *minimal sufficient statistic*. This suggestive nomenclature is explained in [6], [10].

The length of the minimal two-part code for x consisting of the model cost $K(S)$ and the length of the index of x in S, the complexity of S upper bounded by α, is given by the *MDL function*:

$$\lambda_x(\alpha) = \min_S \{\Lambda(S) : S \ni x,\ K(S) \leq \alpha\}, \tag{II.10}$$

where $\Lambda(S) = \log |S| + K(S) \geq K(x) - O(1)$ is the total length of two-part code of x with help of model S. Apart from being convenient for the technical analysis in this work, $\lambda_x(\alpha)$ is the celebrated two-part Minimum Description Length code length (Example IV.16) with the model-code length restricted to at most α.

III. Overview of Results

Background and Related Work: A.N. Kolmogorov [15], [16] proposed the combinatorial non-probabilistic approach to an individual data-to-model relation, two-part codes separating the *structure* or *meaningful information* of a string from meaningless *accidental* features. There is no written version, apart from a few lines [15] which we reproduced in Section I, so we have to rely on oral history of witnesses [9], [4], [12] for more details, and, says Tom Cover [4]: "I remember taking many long hours trying to understand the motivation of Kolmogorov's approach." According to Peter Gács, [9]: "Kolmogorov drew a picture of $h_x(\alpha)$ as a function of α monotonically approaching the diagonal [sufficiency line]. Kolmogorov stated that it was known (proved by L.A. Levin) that in some cases it remained far from the diagonal line till the very end." Leonid A. Levin [12]: "Kolmogorov told me [about] $h_x(i)$ (or its inverse, I am not sure) and asked how this $h(i)$ could behave. I proved that $i + h(i) + O(\log i)$ is monotone but otherwise arbitrary within $O(\sqrt{i})$ accuracy; it stabilizes on $K(x)$ when i exceeds $I(x : \text{Halting})$. (Actually, this expression for accuracy was Kolmogorov's re-wording, I gave it in less elegant but equivalent terms—$O(p \log i)$ where p is the number of "jumps".) I do not remember Kolmogorov defining $\beta_x(i)$ or suggesting anything like your result. I never published anything on the topic because I do not believe strings x with significant $I(x : \text{Halting})$ could exist in the world." (Here "Halting" stands for the infinite binary "halting sequence": the ith bit is 1 iff the ith program for the reference universal prefix machine U halts.)

Remark III.1: Levin's statement [12] quoted above appears to suggest that strings x such that $h_x(i) + i$ stabilizes on $K(x)$ only for large i may exist mathematically but are unlikely to occur in nature, because such x's must have a lot of information about the Halting problem. and hence the analysis of their properties is irrelevant. But the statement in question is imprecise. There are two ways to understand the statement: (i) $h_x(i) + i$ stabilizes on $K(x)$ when i exceeds $I(x : \text{Halting})$ or earlier; or (ii) $h_x(i) + i$ stabilizes on $K(x)$ when i exceeds $I(x : \text{Halting})$ and not earlier. It is not clear what "the information in x about the halting problem" is, since the "Halting problem" is not a finite object and thus the notion of information about Halting needs a special definition. The usual $I(x : \text{Halting}) = K(\text{Halting}) - K(\text{Halting} \mid x)$ doesn't make sense since both $K(\text{Halting})$ and $K(\text{Halting} \mid x)$ are infinite. The expression $I(x : \text{Halting}) = K(x) - K(x \mid \text{Halting})$ looks better provided $K(x \mid \text{Halting})$ is understood as $K(x)$ relativized by the Halting problem. In the latter interpretation of $I(x : \text{Halting})$, case (i) is correct and case (ii) is false. The correctness of (i) is implicit in Theorem V.4. A counter example to (ii): Let p be the halting program of length at most n with the greatest running time. It is easy to show that $K(p)$ is about n, and therefore p is a random string of length about n. As a consequence, the complexity of the minimal sufficient statistic α_0 of p is close to 0. On the other hand $I(p : \text{Halting})$ is about n. Indeed, given the oracle for the Halting problem and n we can find x; hence $I(p : \text{Halting}) = K(p) - K(p \mid \text{Halting}) \geq n - K(n) \geq n - 2\log n$. $\diamondsuit$

Related work on so-called "non-stochastic objects" (where $h_x(\alpha)+\alpha$ drops to $K(x)$ only for large α) is [21], [26], [22], [23], [24]. In 1987, [26], [27], V.V. V'yugin established that, for $\alpha = o(|x|)$, the randomness deficiency function

$\beta_x(\alpha)$ can assume all possible shapes (within the obvious constraints). In the survey [5] of Kolmogorov's work in information theory, the authors preferred to mention $\beta_x(\alpha)$, because it by definition optimizes "best fit," rather than $h_x(\alpha)$ of which the usefulness and meaningfulness was mysterious. But Kolmogorov had a seldom erring intuition: we will show that his original proposal h_x in the proper sense incorporates all desirable properties of $\beta_x(\alpha)$, and in fact is superior. In [3], [6], [5] a notion of "algorithmic sufficient statistics", derived from Kolmogorov's structure function, is suggested as the algorithmic approach to the probabilistic notion of sufficient statistic [7], [6] that is central in classical statistics. The paper [10] investigates the algorithmic notion in detail and formally establishes such a relation. The algorithmic (minimal) sufficient statistic is related in [23], [11] to the "minimum description length" principle [18], [2], [29] in statistics and inductive reasoning. Moreover, [10] observed that $\beta_x(\alpha) \leq h_x(\alpha) + \alpha - K(x) + O(1)$, establishing a one-sided relation between (II.7) and (II.8), and the question was raised whether the converse holds.

This Work: The most fundamental result in this paper is the equality

$$\beta_x(\alpha) = h_x(\alpha) + \alpha - K(x) = \lambda_x(\alpha) - K(x) \qquad \text{(III.1)}$$

which holds within logarithmic additive terms in argument and value. Additionally, every set S that witnesses the value $h_x(\alpha)$ (or $\lambda_x(\alpha)$), also witnesses the value $\beta_x(\alpha)$ (but not vice versa). It is easy to see that $h_x(\alpha)$ and $\lambda_x(\alpha)$ are upper semi-computable (Definition VII.1); but we show that $\beta_x(\alpha)$ is neither upper nor lower semi-computable. A priori there is no reason to suppose that a set that witnesses $h_x(\alpha)$ (or $\lambda_x(\alpha)$) also witnesses $\beta_x(\alpha)$, for *every* α. But the fact that they do, vindicates Kolmogorov's original proposal and establishes h_x's pre-eminence over β_x. The result can be taken as a foundation and justification of common statistical principles in model selection such as maximum likelihood or MDL ([18], [2] and our Examples IV.16 and IV.17). The possible (coarse) shapes of the functions λ_x, h_x and β_x are examined in Section IV. Roughly stated: The structure functions λ_x, h_x and β_x can assume all possible shapes over their full domain of definition (up to additive logarithmic precision in both argument and value). As a consequence, so-called "non-stochastic" strings x for which $h_x(\alpha) + \alpha$ stabilize on $K(x)$ for large α are common. This improves and extends V'yugin's result [26], [27] above; it also improves the independent related result of L.A. Levin [12] above; and, applied to "snooping curves" extends a recent result of V'yugin, [28], in Example IV.14. The fact that λ_x can assume all possible shapes over its full domain of definition establishes the significance of (III.1), since it shows that $\lambda_x(\alpha) \gg K(x)$ indeed happens for some x, α pairs. In that case the more or less easy fact that $\beta_x(\alpha) = 0$ for $\lambda_x(\alpha) = K(x)$ is not applicable, and *a priori* there is no reason for (III.1): Why should minimizing a set containing x plus the set's description length also minimize x's randomness deficiency in the set? But (III.1) shows that it does! In Section V, we exhibit a universal construction for sets realizing $h_x(\alpha)$ for all α. We determined the (fine) details of the function shapes in Section VI. (Non-)computability properties are examined in Section VII, incidentally proving a to our knowledge first natural example, β_x, of a function that is not semi-computable but computable with an oracle for the halting problem. For convenience of the reader we have delegated almost all proofs to Appendix A, and all precise formulations and proofs of the (non)computability and (non)approximability of the structure functions to Appendix B.

All Stochastic Properties of the Data: The result (III.1) shows that the function $h_x(\alpha)$ yields all stochastic properties of data x in the following sense: for every α the class of models of maximal complexity α has a best model with goodness-of-fit determined by the randomness deficiency $\beta_x(\alpha) = h_x(\alpha) + \alpha - K(x)$—the equality being taken up to logarithmic precision. For example, for some value α_0 the minimal randomness deficiency $\beta_x(\alpha)$ may be quite large for $\alpha < \alpha_0$ (so the best model in that class has poor fit), but an infinitessimal increase in model complexity may cause $\beta_x(\alpha)$ to drop to zero (and hence the marginally increased model class now has a model of perfect fit), see Figure 1. Indeed, the structure function quantifies the best possible fit for a model in classes of every complexity.

Validity for Extended Models: Following Kolmogorov we analyzed a canonical setting where the models are finite sets. As Kolmogorov himself pointed out, this is no real restriction: the finite sets model class is equivalent, up to a logarithmic additive term, to the model class of probability density functions, as studied in [21], [10]. The analysis is valid, up to logarithmic additive terms, also for the model class of total recursive functions, as studied in [24]. The model class of *computable probability density functions* consists of the set of functions $P : \{0,1\}^* \rightarrow [0,1]$ with $\sum P(x) = 1$. "Computable" means here that there is a Turing machine T_P that, given x and a positive rational ε, computes $P(x)$ with precision ε. The (prefix-) complexity $K(P)$ of a computable (possibly partial) function P is defined by $K(P) = \min_i\{K(i) : \text{Turing machine } T_i \text{ computes } P\}$. A string x is typical for a distribution P if the randomness deficiency $\delta(x \mid P) = -\log P(x) - K(x \mid P)$ is small. The conditional complexity $K(x \mid P)$ is defined as follows. Say that a function A approximates P if $|A(y,\varepsilon) - P(y)| < \varepsilon$ for every y and every positive rational ε. Then $K(x \mid P)$ is the minimum length of a program that given every function A approximating P as an oracle prints x. Similarly, P is c-optimal for x if $K(P) - \log P(x) \leq K(x) + c$. Thus, instead of the data-to-model code length $\log |S|$ for finite set models, we consider the data-to-model code length $-\log P(x)$ (the Shannon-Fano code). The value $-\log P(x)$ measures also how likely x is under the hypothesis P and the mapping $x \mapsto P_{\min}$ where $P_{\min}$ minimizes $-\log P(x)$ over P with $K(P) \leq \alpha$ is a *maximum likelihood estimator*, see figure 2. Our results thus imply that that maximum likelihood estimator always returns a hypothesis with minimum randomness deficiency.

The model class of *total recursive functions* consists

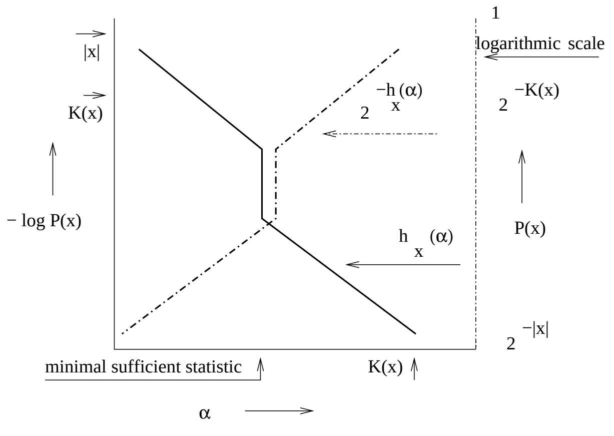

Fig. 2. Structure function $h_x(\alpha) = \min_P\{-\log P(x) : P(x) > 0,\ K(P) \leq \alpha\}$ with P a computable probability density function, with values according to the left vertical coordinate, and the maximum likelihood estimator $2^{-h_x(\alpha)} = \max\{P(x) : P(x) > 0,\ K(P) \leq \alpha\}$, with values according to the right-hand side vertical coordinate.

of the set of computable functions $p : \{0,1\}^* \to \{0,1\}^*$. The (prefix-) complexity $K(p)$ of a total recursive function p is defined by $K(p) = \min_i\{K(i) :$ Turing machine T_i computes $p\}$. In place of $\log |S|$ for finite set models we consider the data-to-model code length $l_x(p) = \min\{|d| : p(d) = x\}$. A string x is typical for a total recursive function p if the randomness deficiency $\delta(x \mid p) = l_x(p) - K(x \mid p)$ is small. The conditional complexity $K(x \mid p)$ is defined as the minimum length of a program that given p as an oracle prints x. Similarly, p is c-optimal for x if $K(p) + l_x(p) \leq K(x) + c$.

It is easy to show that for every data string x and a contemplated finite set model for it, there is an almost equivalent computable probability density function model and an almost equivalent total recursive function model.

Proposition III.2: For every x and every finite set $S \ni x$ there is:

(a) A computable probability density function P with $-\log P(x) = \log |S|$, $\delta(x \mid P) = \delta(x \mid S) + O(1)$ and $K(P) = K(S) + O(1)$; and

(b) A total recursive function p such that $l_x(p) \leq \log |S|$, $\delta(x \mid p) \leq \delta(x \mid S) + O(1)$ and $K(p) = K(S) + O(1)$.

Proof: (a) Define $P(y) = 1/|S|$ for $y \in S$ and 0 otherwise.

(b) If $S = \{x_0, \ldots, x_{m-1}\}$, then define $p(d) = x_{d \bmod m}$. ■

The converse of Proposition III.2 is slightly harder: for every data string x and a contemplated computable probability density function model for it, as well as for a contemplated total recursive function model for x, there is a finite set model for x that has no worse complexity, randomness deficiency, and worst-case data-to-model code for x, up to additive logarithmic precision.

Proposition III.3: There are constants c, C, such that for every string x, the following holds:

(a) For every computable probability density function P there is a finite set $S \ni x$ such that $\log |S| < -\log P(x) + 1$, $\delta(x \mid S) \leq \delta(x \mid P) + 2 \log K(P) + K(\lfloor -\log P(x) \rfloor) + 2 \log K(\lfloor -\log P(x) \rfloor) + C$ and $K(S) \leq K(P) + K(\lfloor -\log P(x) \rfloor) + C$; and

(b) For every total recursive function p there is a finite set $S \ni x$ with $\log |S| \leq l_x(p)$, $\delta(x \mid S) \leq \delta(x \mid p) + 2 \log K(p) + K(l_x(p)) + 2 \log K(l_x(p)) + c$ and $K(S) \leq K(p) + K(l_x(p)) + c$.

Proof: (a) Let $m = \lfloor -\log P(x) \rfloor$, that is, $2^{-m-1} < P(x) \leq 2^{-m}$. Define $S = \{y : P(y) > 2^{-m-1}\}$. Then, $|S| < 2^{m+1} \leq 2/P(x)$, which implies the claimed value for $\log |S|$. To list S it suffices to compute all consecutive values of $P(y)$ to sufficient precision until the combined probabilities exceed $1 - 2^{-m-1}$. That is, $K(S) \leq K(P) + K(m) + O(1)$. Finally, $\delta(x \mid S) = \log |S| - K(x|S^*) < -\log P(x) - K(x \mid S^*) + 1 = \delta(x \mid P) + K(x \mid P) - K(x \mid S^*) + 1 \leq \delta(x \mid P) + K(S^* \mid P) + O(1)$. The term $K(S^* \mid P)$ can be upper bounded as $K(K(S)) + K(m) + O(1) \leq 2 \log K(S) + K(m) + O(1) \leq 2 \log(K(P) + K(m)) + K(m) + O(1) \leq 2 \log K(P) + 2 \log K(m) + K(m) + O(1)$, which implies the claimed bound for $\delta(x \mid S)$.

(b) Define $S = \{y : p(d) = y,\ \ |d| = l_x(p)\}$. Then, $\log |S| \leq l_x(p)$. To list S it suffices to compute $p(d)$ for every argument of length equal $l_x(p)$. Hence, $K(S) \leq K(p) + K(l_x(p)) + O(1)$. The upper bound for $\delta(x \mid S)$ is derived just in the same way as in the proof of item (a). ■

Remark III.4: How large are the nonconstant additive complexity terms in Proposition III.3 for strings x of length n? In item (a), we are commonly only interested in P such that $K(P) \leq n + O(\log n)$ and $-\log P(x) \leq n + O(1)$. Indeed, for every P there is P' such that $K(P') \leq \min\{K(P), n\} + O(\log n)$, $\delta(x \mid P') \leq \delta(x \mid P) + O(\log n)$, $-\log P'(x) \leq \min\{-\log P(x), n\} + 1$. Such P' is defined as follows: If $K(P) > n$ then $P'(x) = 1$ and $P'(y) = 0$ for every $y \neq x$; otherwise $P' = (P + U_n)/2$ where U_n stands for the uniform distribution on $\{0,1\}^n$. Then the additive terms in item (a) are $O(\log n)$. In item (b) we are commonly only interested in p such that $K(p) \leq n + O(\log n)$ and $l_x(p) \leq n + O(1)$. Indeed, for every p there is p' such that $K(p') \leq \min\{K(p), n\} + O(\log n)$, $\delta(x \mid p') \leq \delta(x \mid p) + O(\log n)$, $l_x(p') \leq \min\{l_x(p), n\} + 1$. Such p' is defined as follows: If $K(p) > n$ then p' maps all strings to x; otherwise $p'(0u) = p(u)$ and $p'(1u) = u$. Then the additive terms in item (b) are $O(\log n)$. Thus, in this sense all results in this paper that hold for finite set models extend, up to a logarithmic additive term, to computable probability density function models and to total recursive function models. Since the results in this paper hold only up to additive logarithmic term anyway, this means that all of them equivalently hold for the model class of computable probability density functions, as well as for the model class of total recursive functions. ♢

IV. Coarse Structure

Let $\beta_x(\alpha)$ be defined as in (II.7) and $h_x(\alpha)$ be defined as in (II.8). Both functions are 0 ($\beta_x(\alpha)$ may be $-O(1)$) for all $\alpha \geq K(x) + c_0$ where c_0 is a constant. We *represent* the coarse shape of these functions for different x by functions characteristic of that shape. Informally, g represents

f means that the graph of f is contained in a strip of logarithmic (in the length n of x) width centered on the graph of g, Figure 3.

Intuition: f follows g up to a prescribed precision.

For formal statements we rely on the notion in Definition IV.1. Informally, we obtain the following results (x is of length n and complexity $K(x) = k$):

- Every non-increasing function β represents β_x for some x, and for every x the function β_x is represented by some β, provided $\beta(k) = 0$, $\beta(0) \leq n - k$.
- Every function h, with non-increasing $h(\alpha) + \alpha$, represents h_x for some x, and for every x the function h_x is represented by some h as above, provided $h(k) = 0$, $h(0) \leq n$ (and by the non-increasing property $h(0) \geq k$).
- $h_x(\alpha) + \alpha$ represents $\beta_x(\alpha) + k$, and conversely, for every x.
- For every x and α, every minimal size set $S \ni x$ of complexity at most $\alpha' = \alpha + O(\log n)$, has randomness deficiency $\beta_x(\alpha') \leq \delta(x \mid S) \leq \beta_x(\alpha) + O(\log n)$.

To provide precise statements we need a definition.

Definition IV.1: Let f, g be functions defined on $\{0, 1, \ldots, k\}$ with values in $\mathbb{N} \cup \{\infty\}$. We say that f is $(\varepsilon(i), \delta(i))$-close to g (in symbols: $f = \mathcal{E}(g)$) if

$$f(i) \geq \min\{g(j) \colon j \in [\varepsilon(0), k],\ |j - i| \leq \varepsilon(i)\} - \delta(i)\},$$
$$f(i) \leq \max\{g(j) \colon j \in [\varepsilon(0), k],\ |j - i| \leq \varepsilon(i)\} + \delta(i)\}$$

for every $i \in [\varepsilon(0), k]$. If $f = \mathcal{E}(g)$ and $g = \mathcal{E}(f)$ we write $f \cong g$.

Here $\varepsilon(i), \delta(i)$ are small values like $O(\log n)$ when we consider data x of length n. Note that this definition is not symmetric and allows $f(i)$ to have arbitrary values for $i \in [0, \varepsilon(0))$. However, it is transitive in the following sense: if f is $(\varepsilon_1(i), \delta_1(i))$-close to g and g is $(\varepsilon_2(i), \delta_2(i))$-close to h then f is $(\varepsilon_1(i) + \varepsilon_2(i), \delta_1(i) + \delta_2(i))$-close to h. If $f = \mathcal{E}(g)$ and g is linear continuous, meaning that $|g(i) - g(j)| \leq c|i - j|$ for some constant c, then the difference between $f(i)$ and $g(i)$ is bounded by $c\varepsilon(i) + \delta(i)$ for every $\varepsilon(0) \leq i \leq k$.

This notion of closeness, if applied unrestricted, is not always meaningful. For example, take as g the function taking value n for all even $i \in [0, k]$ and 0 for all odd $i \in [0, k]$. Then for *every* function f on $[0, k]$ with $f(i) \in [0, n]$ we have $f = \mathcal{E}(g)$ for $\varepsilon = 1$, $\delta = 0$. But if $f = \mathcal{E}(g)$ and g is *non-increasing* then g indeed gives much information about f.

It is instructive to consider the following example. Let $g(i)$ be equal to $2k - i$ for $i = 0, 1, \ldots, \frac{k}{2} - 1$ and to $k - i$ for $i = \frac{k}{2}, \ldots, k$. Let $\varepsilon(i), \delta(i)$ be constant. Then a function $f = \mathcal{E}(g)$ may take every value for $i \in [0, \varepsilon)$, every value in $[2k - i - 2\delta, 2k - i + 2\delta]$ for $i \in [\varepsilon, \frac{k}{2} - \delta]$, every value in $[k - i - \delta, 2k - i + \delta]$ for $i \in (\frac{k}{2} - \delta, \frac{k}{2} + \delta]$, and every value in $[k - i - 2\delta, k - i + 2\delta]$ for $i \in (\frac{k}{2} + \delta, k]$ (see Figure 3). Thus the point $\frac{k}{2}$ of discontinuity of g gives an interval of size 2δ of large ambiguity of f. Loosely speaking the graph of f can be any function contained in the strip of radius 2δ whose middle line is the graph of g. For technical reasons it is convenient to use, in place of h_x, the MDL function λ_x (II.10). The definition of λ_x immediately implies the following properties: $\lambda_x(\alpha)$ is non-increasing, $\lambda_x(\alpha) \geq K(x) - O(1)$ for all α.

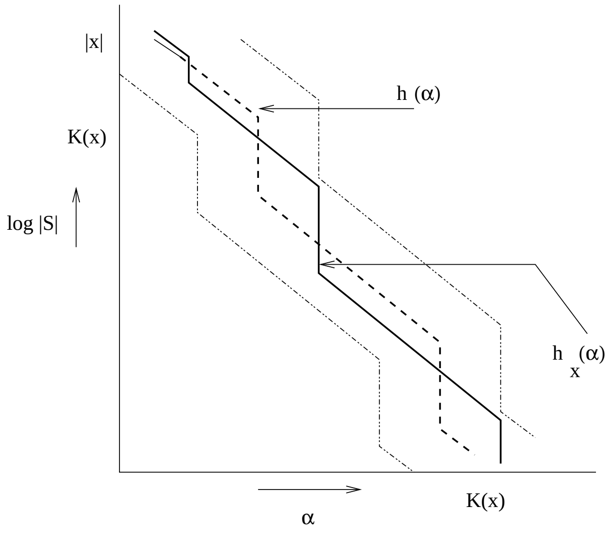

Fig. 3. Structure function $h_x(\alpha)$ in strip determined by $h(\alpha)$, that is, $h_x(\alpha) = \mathcal{E}(h(\alpha))$.

The next lemma shows that properties of λ_x translate directly into properties of h_x since $h_x(\alpha)$ is always "close" to $\lambda_x(\alpha) - \alpha$.

Lemma IV.2: For every x we have $\lambda_x(\alpha) \leq h_x(\alpha) + \alpha \leq \lambda_x(\alpha) + K(\alpha) + O(1)$ for all α. Hence $\lambda_x(\alpha) \cong h_x(\alpha) + \alpha$ for $\varepsilon = 0$, $\delta = K(\alpha) + O(1)$.

Intuition: The functions $h_x(\alpha) + \alpha$ (the ML code length plus the model complexity) and $\lambda_x(\alpha)$ (the MDL code length) are essentially the same function.

Remark IV.3: The lemma implies that the *same set* witnessing $h_x(\alpha)$ also witnesses $\lambda_x(\alpha)$ up to an additive term of $K(\alpha)$. The converse is only true for the smallest cardinality set witnessing $\lambda_x(\alpha)$. Without this restriction a counter example is: for random $x \in \{0,1\}^n$ the set $S = \{0,1\}^n$ witnesses $\lambda_x(\frac{n}{2}) = n + O(K(n))$ but does not witness $h_x(\frac{n}{2}) = \frac{n}{2} + O(K(n))$. (If $\lambda_x(\alpha) = K(x)$, then every set of complexity $\alpha' \leq \alpha$ witnessing $\lambda_x(\alpha') = K(x)$ also witnesses $\lambda_x(\alpha) = K(x)$.) $\diamondsuit$

The next two theorems state the main results of this work in a precise form. By $K(i, n, \lambda)$ we mean the minimum length of a program that outputs n, i, and computes $\lambda(j)$ given any j in the domain of λ. We first analyze the possible shapes of the structure functions.

Theorem IV.4: (i) For every n and every string x of length n and complexity k there is an integer valued non-increasing function λ defined on $[0, k]$ such that $\lambda(0) \leq n$, $\lambda(k) = k$ and $\lambda_x = \mathcal{E}(\lambda)$ for $\varepsilon = \delta = K(n) + O(1)$.

(ii) Conversely, for every n and non-increasing integer valued function λ whose domain includes $[0, k]$ and such that $\lambda(0) \leq n$ and $\lambda(k) = k$, there is x of length n and complexity $k \pm (K(k, n, \lambda) + O(1))$ such that $\lambda_x = \mathcal{E}(\lambda)$ for $\varepsilon = \delta = K(i, n, \lambda) + O(1)$.

Intuition The MDL code length λ_x, and therefore by Lemma IV.2 also the original structure function h_x, can assume essentially every possible shape as a function of the *contemplated maximal model complexity.*

Remark IV.5: The theorem implies that for every function $h(i)$ defined on $[0,k]$ such that the function $\lambda(i) = h(i) + i$ satisfies the conditions of item (ii) there is an x such that $h_x(i) = \mathcal{E}(h(i))$ with $\varepsilon = \delta = O(K(i,n,h))$. ♢

Remark IV.6: The proof of the theorem shows that for every function $\lambda(i)$ satisfying the conditions of item (ii) there is x such that $\lambda_x(i \mid n, \lambda) = \mathcal{E}(\lambda(i))$ with $\varepsilon = \delta = K(i) + O(1)$ where the conditional structure function $\lambda_x(i \mid y) = \min_S\{K(S \mid y) + \log|S| : S \ni x,\ K(S \mid y) \le i\}$. Consequently, for every function $h(i)$ such that the function $\lambda(i) = h(i) + i$ satisfies the conditions of item (ii) there is an x such that $h_x(i \mid n, h) = \mathcal{E}(h(i))$ with $\varepsilon = \delta = O(K(i))$ where the conditional structure function $h_x(i \mid y) = \min_S\{\log|S| : S \ni x,\ K(S \mid y) \le i\}$. ♢

Remark IV.7: In the proof of Item (ii) of the theorem we can consider every finite set U with $|U| \ge 2^n$ in place of the set A of all strings of length n. Then we obtain a string $x \in U$ such that $\lambda_x = \mathcal{E}(\lambda)$ with $\varepsilon(i) = \delta(i) = K(i, U, \lambda)$. ♢

The following central result of this paper shows that the λ_x (equivalently h_x, by Lemma IV.2) and β_x can be expressed in one another but for a logarithmic additive error.

Theorem IV.8: For every x of length n and complexity k it holds $\beta_x(\alpha) + k \cong \lambda_x(\alpha)$ for $\varepsilon = \delta = O(\log n)$.

Intuition: A model achieving the MDL code length $\lambda_x(\alpha)$, or the ML code length $h_x(\alpha)$, essentially achieves the best possible fit $\beta_x(\alpha)$.

Corollary IV.9: For every x of length n and complexity $k \le n$ there is a non-increasing function β such that $\beta(0) \le n - k$, $\beta(k) = 0$ and $\beta_x = \mathcal{E}(\beta)$ for $\varepsilon, \delta = O(\log n)$. Conversely, for every non-increasing function β such that $\beta(0) \le n-k$, $\beta(k) = 0$ there is x of length n and complexity $k \pm \delta$ such that $\beta_x = \mathcal{E}(\beta)$ for $\varepsilon = \delta = O(\log n) + K(\beta)$.

Proof: The first part is more or less immediate. Or use the first part of Theorem IV.4 and then let $\beta(i) = \lambda(i) - k$. To prove the second part, use the second part of Theorem IV.8, and the second part of Theorem IV.4 with $\lambda(i) = \beta(i) + k$. ■

Remark IV.10: From the proof of Theorem IV.8 we see that for every finite set $S \ni x$, of complexity at most $\alpha + O(\log n)$ and minimizing $\Lambda(S)$, we have $\delta(x \mid S) \le \beta_x(\alpha) + O(\log n)$. Ignoring $O(\log n)$ terms, at every complexity level, every best hypothesis at this level with respect to $\Lambda(S)$ is also a best one with respect to typicality. This explains why it is worthwhile to find shortest two-part descriptions for given data x: this is the single known way to find an $S \ni x$ with respect to which x is as typical as possible at that complexity level. Note that the set $\{\langle x, S, \beta\rangle \mid x \in S,\ \delta(x \mid S) < \beta\}$ is not enumerable so we are not able to generate such S's directly (Section VII).

The converse is not true: not every hypothesis, consisting of a finite set, witnessing $\beta_x(\alpha)$ also witnesses $\lambda_x(\alpha)$ or $h_x(\alpha)$. For example, let x be a string of length n with $K(x) \ge n$. Let $S_1 = \{0,1\}^n \cup \{y\}$ where y is a string of length $\frac{n}{2}$ such that $K(x,y) \ge \frac{3n}{2}$ and let $S_2 = \{0,1\}^n$. Then both S_1, S_2 witness $\beta_x(\frac{n}{2} + O(\log n)) = O(1)$ but $\Lambda(S_1) = \frac{3n}{2} + O(\log n) \gg \lambda_x(\frac{n}{2} + O(\log n)) = n + O(\log n)$ while $\log|S_2| = n \gg h_x(\frac{n}{2} + O(\log n)) = \frac{n}{2} + O(\log n)$. ♢

However, for every α such that $\lambda_x(i)$ decreases when $i \to \alpha$ with $i \le \alpha$, a witness set for $\beta_x(\alpha)$ is also a witness set for $\lambda_x(\alpha)$ and $h_x(\alpha)$. We will call such α *critical* (with respect to x): these are the model complexities at which the two-part MDL code-length decreases, while it is stable in between such critical points. The next theorem shows, for critical α, that for every $A \ni x$ with $K(A) \approx \alpha$ and $\delta(x \mid A) \approx \beta_x(\alpha)$, we have $\log|A| \approx h_x(\alpha)$ and $\Lambda(A) \approx \lambda_x(\alpha)$. More specifically, if $K(A) \approx \alpha$ and $\delta(x \mid A) \approx \beta_x(\alpha)$ but $\Lambda(A) \gg \lambda_x(\alpha)$ or $\log|A| \gg h_x(\alpha)$ then there is $S \ni x$ with $K(S) \ll \alpha$ and $\Lambda(S) \approx \lambda_x(\alpha)$.

Theorem IV.11: For all $A \ni x$ there is $S \ni x$ such that $\Lambda(S) \le \lambda_x(\alpha) + (\delta(x|A) - \beta_x(\alpha))$, $K(S) \le K(A) + (\lambda_x(\alpha) - \Lambda(A)) + (\delta(x|A) - \beta_x(\alpha))$, and $K(S) \le \alpha + (h_x(\alpha) - \log|A|) + (\delta(x|A) - \beta_x(\alpha))$ where all inequalities hold up to $O(\log \Lambda(A))$ additive term.

Intuition: Although models of best fit (witnessing $\beta_x(\alpha)$) do not necessarily achieve the MDL code length $\lambda_x(\alpha)$ or the ML code length h_x, they do so at the model complexities where the MDL code length decreases, and, equivalently, the ML code length decreases at a slope of more than -1.

Remark IV.12: **Invariance under Recoding of Data:** In what sense is the structure function invariant under recoding of the data? Osamu Watanabe suggested the example of replacing the data x by a shortest program x^* for it. Since x^* is incompressible it is a typical element of the set of all strings of length $|x^*| = K(x)$, and hence $h_{x^*}(\alpha)$ drops to the sufficiency line $L(\alpha) = K(x) - \alpha$ already for some $\alpha \le K(K(x))$, so almost immediately (and it stays within logarithmic distance of that line henceforth). That is, $h_{x^*}(\alpha) = K(x) - \alpha$ up to logarithmic additive terms in argument and value, irrespective of the (possibly quite different) shape of h_x. We note that the recoding function $f(x) = x^*$ is not recursive since $K(x^*|x) \ge \log|x| - \log\log|x|$ for some x of each length by [8], and, while it is one-one and total the function is not onto. But it is the partiality of the inverse function (not all strings are shortest programs) that causes the collapse of the structure function. If one restricts the finite sets containing x^* to be subsets of $\{y^* : y \in \{0,1\}^*\}$, then the resulting structure function h_{x^*} is within a logarithmic strip around h_x. However, the structure function is invariant under "proper" recoding of the data.

Lemma IV.13: Let f be a recursive permutation of the set of finite binary strings (one-one, total, and onto). Then, $h_{f(x)} = \mathcal{E}(h_x)$ for $\varepsilon, \delta = K(f) + O(1)$.

Proof: Let $S \ni x$ be a witness of $h_x(\alpha)$. Then, $S_f = \{f(y) : y \in S\}$ satisfies $K(S_f) \le \alpha + K(f) + O(1)$ and $|S_f| = |S|$. Hence, $h_{f(x)}(\alpha + K(f) + O(1)) \le h_x(\alpha)$. Let $R \ni f(x)$ be a witness of $h_{f(x)}(\alpha)$. Then, $R_{f^{-1}} = \{f^{-1}(y) : y \in R\}$ satisfies $K(R_{f^{-1}}) \le \alpha + K(f) + O(1)$ and $|R_{f^{-1}}| = |R|$. Hence, $h_x(\alpha + K(f) + O(1)) \le h_{f(x)}(\alpha)$ (since $K(f^{-1}) = K(f) + O(1)$). ■

♢

Example IV.14: **Best Prediction Strategy:** In [28] the notion of a *snooping curve* $L_x(\alpha)$ of x was introduced, expressing the minimal logarithmic loss in predicting the consecutive elements of a given individual string x, in each prediction using the preceding sequence of elements, by the best prediction strategy of complexity at most α.

Intuition: The snooping curve quantifies the quality of the best predictor for a given sequence at every possible predictor-complexity.

Formally, $L_x(\alpha) = \min_{K(P)\leq\alpha} \mathrm{Loss}_P(x)$. The minimum is taken over all prediction strategies P of complexity at most α. A prediction strategy P is a mapping from the set of strings of length less than $|x|$ into the set of rational numbers in the segment $[0,1]$. The value $P(x_1 \ldots x_i)$ is regarded as our belief (or probability) that $x_{i+1} = 1$ after we have observed $x_1, \ldots, x_i$. If the actual bit x_{i+1} is 1 the strategy suffers the loss $-\log p$ otherwise $-\log(1-p)$. The strategy is a finite object and $K(P)$ may by defined as the complexity of this object, or as the minimum size of a program that identifies $n = |x|$ and given y finds $P(y)$. The notation $\mathrm{Loss}_P(x)$ indicates the total loss of P on x, i.e. the sum of all n losses: $\mathrm{Loss}_P(x) = \sum_{i=0}^{|x|-1}(-\log|P(x_1 \ldots x_i) - 1 + x_{i+1}|)$. Thus, the snooping curve $L_x(\alpha)$ gives the minimal loss suffered on all of x by a prediction strategy, as a function of the complexity at most α of the contemplated class of prediction strategies. The question arises what shapes these functions can have—for example, whether there can be sharp drops in the loss for only minute increases in complexity of prediction strategies.

A result of [28] describes possible shapes of L_x but only for $\alpha = o(n)$ where n is the length of x. Here we show that for every function L and every $k \leq n$ there is a data sequence x such that $L_x(\alpha \pm O(\log n)) = L(\alpha) \pm O(\log n)$, provided $L(0) \leq n$, $L(\alpha)+\alpha$ is non-increasing on $[0,k]$, and $L(\alpha) = 0$ for $\alpha \geq k$.

Lemma IV.15: $L_x(\alpha \pm O(\log n)) = h_x(\alpha \pm O(\log n))$ for every x and α. Thus, Lemma IV.2 and Theorem IV.4 describes also the coarse shape of all possible snooping curves.

Proof: ($\leq$) A given finite set A of binary strings of length n can be identified with the following prediction strategy P: Having read the prefix y of x it outputs $p = |A_{y1}|/|A_y|$ where A_y stands for the number of strings in A having prefix y.

It is easily seen, by induction, that $\mathrm{Loss}_P(y) = \log(|A|/|A_y|)$ for every y. Therefore, $\mathrm{Loss}_P(x) = \log|A|$ for every $x \in A$. Since P corresponds to A in the sense that $K(P \mid A) = O(1)$, we obtain $L_x(\alpha + O(\log n)) \leq h_x(\alpha)$. The term $O(\log n)$ is required, because the initial set of complexity α might contain strings of different lengths while we need to know n to get rid of the strings of lengths different from n.

($\geq$) Conversely, assume that $\mathrm{Loss}_P(x) \leq m$. Let $A = \{x \in \{0,1\}^n \colon \mathrm{Loss}_P(x) \leq m\}$. Since $\sum_{|x|=n} 2^{-\mathrm{Loss}_P(x)} = 1$ (proof by induction on n), and $2^{-\mathrm{Loss}_P(x)} \geq 2^{-m}$ for every $x \in A$, we can conclude that A has at most 2^m elements. Since $K(A \mid P) = O(\log m)$, we obtain $h_x(\alpha + O(\log n)) \leq L_x(\alpha)$. ■

Thus, within the obvious constraint of the function $L_x(\alpha) + \alpha$ being non-increasing, all shapes for the minimal total loss $L_x(\alpha)$ as a function of the allowed predictor complexity are possible. ♢

Example IV.16: **Foundations of MDL:** (i) Consider the following algorithm based on the Minimum Description Length principle. Given x, the data to explain, and α, the maximum allowed complexity of explanation, we search for programs p of length at most α that print a finite set $S \ni x$. Such pairs (p,S) are possible explanations. The *best explanation* is defined to be the (p,S) for which $\delta(x|S)$ is minimal. Since the function $\delta(x|S)$ is not computable, we cannot find the best explanation. The programs use unknown computation time and thus we can never be certain that we have found all possible explanations.

To overcome this problem we use the indirect method of MDL: We run all programs in dovetailed fashion. At every computation step t consider all pairs (p,S) such that program p has printed the set S containing x by time t. Let (p_t, L_t) stand for the pair (p,S) such that $|p| + \log|S|$ is minimal among all these pairs (p,S). The best hypothesis L_t changes from time to time due to the appearance of a better hypothesis. Since no hypothesis is declared best twice, from some moment onwards the explanation (p_t, L_t) which is declared best does not change anymore.

Compare this indirect method with the direct one: after step t of dovetailing select (p,S) for which $\log|S| - K^t(x|S)$ is minimum among all programs p that up to this time have printed a set S containing x, where $K^t(x|S)$ is the approximation of $K^t(x|S)$ obtained after t steps of dovetailing, that is, $K^t(x|S) = \min\{|q| :$ U on input $\langle q, S\rangle$ prints x in at most t steps$\}$. Let (q_t, B_t) stand for that model. This time the same hypothesis can be declared best twice. However from some moment onwards the explanation (q_t, B_t) which is declared best does not change anymore.

Why do we prefer the indirect method to the direct one? The explanation is that in practice we deal often with t that are much less than the time of stabilization of both L_t and B_t. For small t, the model L_t is better than B_t in the following respect: L_t has some guarantee of goodness, as we know that $\delta(x|L_t) + K(x) \leq |p_t| + \log|L_t| + O(1)$. That is, we know that the sum of deficiency of x in L_t and $K(x)$ is less than some known value. In contrast, the model B_t has no guarantee of goodness at all: we do not know any upper bound neither for $\delta(x|B_t)$, nor for $\delta(x|B_t) + K(x)$.

Theorem IV.8 implies that the indirect method of MDL gives not only some garantee of goodness but also that, in the limit, that guarantee approaches the value it upper bounds, that is, approaches $\delta(x|L_t) + K(x)$, and $\delta(x|L_t)$ itself is not much greater than $\delta(x|B_t)$ (assuming that α is not critical). That is, in the limit, the method of MDL will yield an explanation that is only a little worse than the best explanation.

(ii) If $S \ni x$ is a smallest set such that $K(S) \leq \alpha$, then S can be converted into a best strategy of complexity at most α, to predict the successive bits of x given the preceding ones, (Example IV.14). Interpreting "to explain" as "to be able to predict well", MDL in the sense of sets witnessing $\lambda_x(\alpha)$ gives indeed a good explanations at every complexity level α.

(iii) In statistical applications of MDL [18], [2], MML [29], and related methods, one selects the model in a given model class that minimizes the sum of the model code length and the data-to-model code length; in modern versions one selects the model that minimizes just the data-to-model code length (ignoring the model code length). For example, one uses data-to-model code $-\log P(x)$ for data x with respect to probability (density function) model P. For example, if the model is the uniform distribution over n-bit strings, then the data-to-model code for $x = 00\ldots0$ is $-\log 1/2^n = n$, even though we can compress x to about $logn$ bits, without even using the model. Thus, the data-to-model code is the worst-case number of bits required for data of given length using the model, rather than the optimal number of bits for the particular data at hand. This is precisely what we do in the structure function approach: the data-to-model cost of x with respect to model $A \ni x$ is $\log|A|$, the worst-case number of bits required to specify an element of A rather than the minimal number of bits required to specify x in particular. In contrast, ultimate compression of the two-part code, which is suggested by the "minimum description length" phrase, [23], means minimizing $K(A)+K(x|A)$ over all models A in the model class. In Theorem IV.8 we have essentially shown that the "worst-case" data-to-model code above is the approach that guarantees the best fitting model. In contrast, the "ultimate compression" approach can yield models that are far from best fit. (It is easy to see that this happens only if the data are "not typical" for the contemplated model, [23].) For instance, let x be a string of length n and complexity about $n/2$ for which $\beta_x(O(\log(n)) = n/4 + O(\log(n)$. Such string exists by Corollary IV.9. And let the model class consists of all finite sets containing x of complexity at most $\alpha = O(\log n)$. Then for the model $A_0 = \{0,1\}^n$ we have $K(A_0) = O(\log n)$ and $K(x|A_0) = n/2+O(\log n)$ thus the sum $K(A_0)+K(x|A_0) = n/2+O(\log n)$ is minimal up to a term $O(\log n)$. However, the randomness defficiency of x in A_0 is about $n/2$, which is much bigger than the minimum $\beta_x(O(\log(n)) \approx n/4$. For the model A_1 witnessing $\beta_x(O(\log(n)) \approx n/4$ we also have $K(A_1) = O(\log n)$ and $K(x|A_1) = n/2 + O(\log n)$. However, it has smaller cardinality: $\log|A_1| = 3n/4 + O(\log n)$ which causes the smaller randomness deficiency.

The same happens also for other model classes, such as probability models. Consider, for instance, the class of Bernoulli processes with rational bias p for outcome "1" $(0 \le p \le 1)$ to generate binary strings of length n. Suppose we look for the model minimizing the codelength of the model plus data given the model: $K(p|n) + K(x|p,n)$. Let the data be $x = 00\ldots0$. Then the model corresponding to probability $p = \frac{1}{2}$ compresses the data code to $K(x \mid n,p) = O(1)$ bits and $K(p|n) = O(1)$. But we find about the same code length if we take $p' = 0$. Thus we have no basis to distinguish between the two, while obviously the second possibility is preferable. This shows that ultimate compression of the two-part code, here resulting in $K(p|n) + K(x|n,p)$, may yield a (probability) model P for which the data has large randomness deficiency ($-\log P(x) - K(x \mid n,p) = n$ for $p = \frac{1}{2}$) and hence is atypical.

However, in the structure functions $h_x(\alpha)$ and $\lambda_x(\alpha)$ the data-to-model code for the model $p = \frac{1}{2}$ is $-\log P(x) = -\log(\frac{1}{2})^n = n$ bits, while $p = 0$ results $-\log 1^n = 0$ bits. Choosing the shortest data-to-model code results in the minimal randomness deficiency, as in (the generalization to probability distributions of) Theorem IV.8.

(iv) Another question arising in MDL or maximum likelihood (ML) estimation is its performance if the "true" model is not part of the contemplated model class. Given certain data, why would we assume they are generated by probabilistic or deterministic processes? They have arisen by natural processes most likely not conforming to mathematical idealization. Even if we can assume the data arose from a process that can be mathematically formulated, such situations arise if we restrict modeling of data arising from a "complex" source (conventional analogue being data arising from $2k$-parameter sources) by "simple" models (conventional analogue being k-parameter models). Again, Theorem IV.8 shows that, within the class of models of maximal complexity α, these constraints we still select a simple model for which the data is maximally typical. This is particularly significant for data x if the allowed complexity α is significantly below the complexity of the Kolmogorov minimal sufficient statistic, that is, if $h_x(\alpha) + \alpha \gg K(x) + c$. This situation is potentially common, for example if we have a small data sample generated by a complex process. Then, the data will typically be non-stochastic in the sense of Example IV.20. For a data sample that is very large relative to the complexity of the process generating it, this will typically not be the case and the structure function will drop to the sufficiency line early on. $\diamondsuit$

Example IV.17: **Foundations of Maximum Likelihood:** The algorithm based on ML principle is similar to the algorithm of the previous example. The only difference is that the currently best (p,S) is the one for which $\log|S|$ is minimal. In this case the limit hypothesis $\tilde{S}$ will witness $h_x(\alpha)$ and we obtain the same corollary: $\delta(x|S) \le \beta_x(\alpha - O(\log n)) + O(\log n)$. $\diamondsuit$

Example IV.18: **Approximation Improves Models:** Assume that in the MDL algorithm, as described in Example IV.16, we change the currently best explanation (p_1, S_1) to the explanation (p_2, S_2) only if $|p_2|+\log|S_2|$ is much less than $|p_1|+\log|S_1|$, say $|p_2|+\log|S_2| \le |p_1|+\log|S_1|-c\log n$ for a constant c. It turns out that if c is large enough and p_1 is a shortest program of S_1, then $\delta(x \mid S_2)$ is much less than $\delta(x \mid S_1)$. That is, every time we change the explanation we improve its goodness unless the change is just caused by the fact that we have not yet found the minimum length program for the current model.

Lemma IV.19: There is a constant c such that if $\Lambda(S_2) \le \Lambda(S_1) - 2c\log|x|$, then $\delta(x \mid S_2) \le \delta(x \mid S_2) - c\log|x| + O(1)$.

Proof: Assume the notation of Theorem IV.8. By (A.4), for every pair of sets $S_1, S_2 \ni x$ we have $\delta(x \mid S_2) -$

$\delta(x \mid S_1) = \Lambda(S_2) - \Lambda(S_1) + \Delta$ with $\Delta = K(S_1 \mid x^*) - K(S_2 \mid x^*) + O(1) \leq K(S_1 \mid S_2, x^*) + O(1) \leq K(S_1 \mid S_2, x) + O(1)$. As $\Lambda(S_2) - \Lambda(S_1) \leq |p_2| + \log|S_2| - \Lambda(S_1) = |p_2| + \log|S_2| - (|p_1| + \log|S_1|) \leq -2c\log|x|$ we need to prove that $K(S_2 \mid S_1, x) \leq c\log|x| + O(1)$. Note that (p_1, S_1), (p_2, S_2) are consecutive explanations in the algorithm and every explanation may appear only once. Hence to identify S_1 we only need to know p_2, S_2, α and x. Since p_2 may be found from S_2 and length $|p_2|$ as the first program computing S_2 of length $|p_2|$, obtained by running all programs dovetailed style, we have $K(S_2 \mid S_1, x) \leq 2\log|p_2| + 2\log|\alpha| + O(1) \leq 4\log|x| + O(1)$. Hence we can choose $c = 4$. (Continued in Example VI.5.) ■

◇

Example IV.20: **Non-stochastic objects:** Let α_0, β_0 be natural numbers. A string x is called (α_0, β_0)-*stochastic* by Kolmogorov if $\beta_x(\alpha_0) \leq \beta_0$. In [21] it is proven that for some c, C for all n and all α_0, β_0 with $2\alpha_0 + \beta_0 < n - c\log n - C$ there is a string x of length n that is not (α_0, β_0)-stochastic. Corollary IV.9 strengthens this result of Shen: for some c, C for all n and all α_0, β_0 with $\alpha_0 + \beta_0 < n - c\log n - C$ there is a string x of length n that is not (α_0, β_0)-stochastic. Indeed, apply Corollary IV.9 to $k = \alpha_0 + c_1 \log n + C_1$ (we will choose c_1, C_1 later) and the function $\beta(i) = n - k$ for $i < k$ and $\beta(i) = 0$ for $i = k$. For the x existing by Corollary IV.9 we have $\beta_x(\alpha_0) \geq \beta(\alpha_0 \pm (c_2\log n + C_2)) - (c_2 \log n + C_2) \geq \beta(k-1) - (c_2\log n + C_2) = n - k - (c_2 \log n + C_2) = n - (\alpha_0 + c_1 \log n + C_1) - (c_2 \log n + C_2) > \beta_0$. (The first inequality is true if $\alpha_0 + c_2 \log n + C_2 \leq k - 1$; thus let $c_1 = c_2, C_1 = C_2 + 1$. For the last inequality to be true let $c = c_1 + c_2$ and $C = C_1 + C_2$.) That is, x is not (α_0, β_0)-stochastic. ◇

V. Realizing the Structure Function

We give a general construction of the finite sets witnessing λ_x, h_x, and β_x, at each argument (that is, level of model complexity), in terms of indexes of x in the enumeration of strings of given complexity, up to the "coarse" equivalence precision of Section IV. This extends a technique introduced in [10].

Intuition: The question arises whether there is a uniform construction to obtain the models that realize the structure functions at given complexities. Here we present such a construction. (The construction is of course not computable.)

Definition V.1: Let N^l denote the number of strings of complexity at most l, and let $|N^l|$ denote the length of the binary notation of N^l. For $i \leq |N^l|$ let N^l_i stand for i most significant bits of binary notation of N^l. Let D denote the set of all pairs $\{\langle x, l\rangle \mid K(x) \leq l\}$. Fix an enumeration of D and denote by I^l_x the minimum index of a pair $\langle x, i\rangle$ with $i \leq l$ in that enumeration, that is, the number of pairs enumerated before $\langle x, i\rangle$ (if $K(x) > l$ then $I^l_x = \infty$). Let m^l_x denote the maximal common prefix of binary notations of I^l_x and N^l, that is, $I^l_x = m^l_x 0{*}{*}\cdots{*}$ and $N^l = m^l_x 1{*}{*}\cdots{*}$ (we assume here that binary notation of I^l_x is written in exactly $|N^l|$ bits with leading zeros if necessary).

(In [10] the notation m_x is used for m^l_x with $l = K(x)$.)

Theorem V.2: For every $i \leq l$, the number N^l_i is algorithmically equivalent to N^i, that is, $K(N^i \mid N^l_i), K(N^l_i \mid N^i) = O(\log l)$.

Before proceeding to the main theorem of this section we introduce some more notation.

Definition V.3: For $i \leq l$ let S^l_i denote the set of all strings y such that the binary notation of I^l_y has the form $N^l_i 0{*}{*}\cdots{*}$ (we assume here that binary notations of indexes are written using exactly $|N^l|$ bits.)

Let c denote a constant such that $K(x) \leq \Lambda(S) + c$ for every $x \in S$. The following theorem shows that sets S^l_i form a universal family of statistics for x.

Theorem V.4: (i) If the $(i+1)$st most significant bit of N^l is 1 then $|S^l_i| = 2^{|N^l|-i-1}$ and S^l_i is algorithmically equivalent to N^l_i, that is $K(N^l_i \mid S^l_i), K(S^l_i \mid N^l_i) = O(\log l)$.

(ii) For every S and every $x \in S$, let $l = \Lambda(S) + c$ and $i = |m^l_x|$. Then $x \in S^l_i$, $K(S^l_i \mid S) = O(\log l)$, $K(S^l_i) = i + O(\log l) \leq K(S) + O(\log l)$, and $\Lambda(S^l_i) \leq \Lambda(S) + O(\log l)$ (that is, S^l_i is not worse than S, as a model explaining x).

(iii) If α is critical then every S witnessing $\lambda_x(\alpha)$ is algorithmically equivalent to N^α. That is, if $K(S) \approx \alpha$ and $\Lambda(S) \approx \lambda_x(\alpha)$ but $K(N^\alpha|S) \gg 0$ or $K(S|N^\alpha) \gg 0$ then there is $A \ni x$ with $K(A) \ll \alpha$ and $\Lambda(A) \approx \lambda_x(\alpha)$. More specifically, for all $S \ni x$ either $K(S|N^\alpha) \leq K(S) - \alpha$ and $K(N^\alpha|S) = 0$, or there is $A \ni x$ such that $\Lambda(A) \leq \Lambda(S)$ and $K(A) \leq \min\{\alpha - K(N^\alpha|S), K(S) - K(S|N^\alpha)\}$, where all inequalities hold up to $O(\log \Lambda(S))$ additive term.

Note that Item (iii) of the theorem does not hold for noncritical points. For instance, for a random string x of length n there are independent S_1, S_2 witnessing $\lambda_x(\frac{n}{2}) = n$: let S_1 be the set of all x' of length n having the same prefix of length $\frac{n}{2}$ as x and S_2 be the set of all x' of length n having the same suffix of length $\frac{n}{2}$ as x.

Corollary V.5: Let x be a string of length n and complexity k. For every α $(K(n) + O(1) \leq \alpha \leq k)$ there is $l \leq n + K(n) + O(1)$ such that the set S^l_α both contains x and witnesses $h_x(\alpha)$, $\lambda_x(\alpha)$, and $\beta_x(\alpha)$, up to an $O(\log n)$ additive term in the argument and value.

VI. Fine Structure and Sufficient Statistic

Above, we looked at the coarse shape of the structure function, but not at the fine detail. We show that h_x coming from infinity drops to the sufficiency line L defined by $L(\alpha) + \alpha = K(x)$. It first touches this line for some $\alpha_0 \leq K(x) + O(1)$. It then touches this line a number of times (bounded by a universal constant) and in between moves slightly (logarithmically) away in little bumps. There is a simple explanation why these bumps are there: It follows from (II.3) and (II.5) that there is a constant c_1 such that for every $S \ni x$, we have $K(S) + \log|S| \geq K(x) + K(S \mid x^*) - c_1$. If, moreover, $K(S) + \log|S| \leq K(x) + c_2$, then $K(S \mid x^*) \leq c_2 + c_1$. This was already observed in [10]. Consequently, there are less than $2^{c_2+c_1+1}$ distinct such sets S. Suppose the graph of h_x drops within distance c_2 of the sufficiency line at α_0, then it cannot be within distance c_2 on more than $2^{c_2+c_1+1}$ points.

By the pigeon-hole principle, there is $\alpha \in [\alpha_0, K(x)]$ such that $h_x(\alpha) + \alpha \geq \lambda_x(\alpha) \geq K(x) + \log(K(x) - \alpha_0) - c_2 - 1$. So if $|K(x) - \alpha_0|$ is of order $\Omega(n)$, then we obtain the logarithmic bumps, or possibly only one logarithmic bump, on the interval $[\alpha_0, K(x)]$. However, we will show below that h_x cannot move away more than $O(\log |K(x) - \alpha_0|)$ from the sufficiency line on the interval $[\alpha_0, K(x)]$. The intuition here is that a data sequence can have a simple satisfactory probabilistic explanation, but we can also explain it by many only slightly more complex explanations that are slightly less satisfactory but also model more accidental random features—models that are only slightly more complex but that significantly overfit the data sequence by modeling noise.

Initial behavior: Let x be a string of complexity $K(x) = k$. The structure function $h_x(\alpha)$ defined by (II.8) rises sharply above the sufficiency line for very small values of α. Define

$$m(x) = \min_y \{K(y) : y \geq x\}, \qquad \text{(VI.1)}$$

the minimum complexity of a string greater than x—that is, $m(x)$ is the greatest monotonic non-decreasing function that lower bounds $K(x)$. The function $m(x)$ tends to infinity as x tends to infinity, very slowly—slower than any computable function.

For $\alpha \in [0, m(x) - O(1))$ we have $h_x(\alpha) = \infty$: For a set $S \ni x$ with $K(S) = \alpha$ we can consider the largest element y of S. Then y has complexity $\alpha + O(1) < m(x)$, that is, $K(y) < m(x)$, which implies that $y < x$. But then $x \notin S$ which is a contradiction.

Sufficient Statistic: A sufficient statistic of the data contains all information in the data about the model. In introducing the notion of sufficiency in classical statistics, Fisher [7] stated: "The statistic chosen should summarize the whole of the relevant information supplied by the sample. This may be called the Criterion of Sufficiency ... In the case of the normal curve of distribution it is evident that the second moment is a sufficient statistic for estimating the standard deviation." For the classical (probabilistic) theory see, for example, [6]. In [10] an algorithmic theory of sufficient statistic (relating individual data to individual model) was developed and its relation with the probabilistic version established. The algorithmic basics are as follows: Intuitively, a model expresses the essence of the data if the two-part code describing the data consisting of the model and the data-to-model code is as concise as the best one-part description. Formally:

Definition VI.1: A finite set S containing x is *optimal for x* if

$$\Lambda(S) \leq K(x) + c. \qquad \text{(VI.2)}$$

Here c is some small value, constant or logarithmic in $K(x)$, depending on the context. A minimal length description S^* of such an optimal set is called a *sufficient statistic for x*. To specify the value of c we will say *c-optimal* and *c-sufficient*.

If a set S is c-optimal with c constant, then by (II.9) we have $K(x) - c_2 \leq \Lambda(S) \leq K(x) + c$. Hence, with respect to the structure function $\lambda_x(\alpha)$ we can state that all optimal sets S and only those, cause the function λ_x to drop to its minimal possible value $K(x)$. We know that this happens for at least one set, $\{x\}$ of complexity $K(x) + O(1)$.

We are interested in finding optimal sets that have low complexity. Those having minimal complexity are called *minimal optimal sets* (and their programs *minimal sufficient statistics*). To be rigorous we should say minimal among c-optimal. We know from [10] that the complexity of a minimal optimal set is at least $K(K(x))$, up to a fixed additive constant, for every x. So for smaller arguments the structure function definitively rises above the sufficiency line. We also know that for every n there are so-called *non-stochastic* objects x of length n that have optimal sets of high complexity only. For example, there are x of complexity $K(x \mid n^*) = n + O(1)$ such that every optimal set S has also complexity $K(S \mid n^*) = n + O(1)$, hence by the conditional version $K(S \mid n^*) + \log |S| \leq K(x \mid n^*) + c$ of (VI.2) we find $|S|$ is bounded by a fixed universal constant. As $K(S \mid x^*) = O(1)$ (this is proven in the beginning of this section), for every $y \in S$ we have $K(y \mid x^*) \leq K(y \mid S) + K(S \mid x^*) + O(1) = O(1)$. Roughly speaking for such x there is no other optimal set S than the singleton $\{x\}$.

Example VI.2: **Bumps in the Structure Function:** Consider $x \in \{0,1\}^n$ with $K(x \mid n) = n + O(1)$ and the conditional variant $h_x(\alpha \mid y) = \min_S \{\log |S| : S \ni x,\ |S| < \infty,\ K(S \mid y) \leq \alpha\}$ of (II.8). Since $S_1 = \{0,1\}^n$ is a set containing x and can be described by $O(1)$ bits (given n), we find $h_x(\alpha \mid n) \leq n + O(1)$ for $\alpha = K(S_1 \mid n) = O(1)$. For increasing α, the size of a set $S \ni x$, one can describe in α bits, decreases monotonically until for some α_0 we obtain a first set S_0 witnessing $h_x(\alpha_0 \mid n) + \alpha_0 = K(x \mid n) + O(1)$. Then, S_0 is a minimal-complexity optimal set for x, and S_0^* is a minimal sufficient statistic for x. Further increase of α halves the set S for each additional bit of α until $\alpha = K(x \mid n)$. In other words, for every increment d we have $h_x(\alpha_0 + d \mid n) = K(x \mid n) - (\alpha_0 + d + O(\log d))$, provided the right-hand side is non-negative, and 0 otherwise. Namely, once we have an optimal set S_0 we can subdivide it in a standard way into 2^d parts and take as new set S the part containing x. The $O(\log d)$ term is due to the fact that we have to consider self-delimiting encodings of d. This additive term is there to stay, it cannot be eliminated. For $\alpha \geq K(x \mid n)$ obviously the smallest set S containing x that one can describe using α bits (given n) is the singleton set $S = \{x\}$. The same analysis can be given for the unconditional version $h_x(\alpha)$ of the structure function, which behaves the same except for possibly the small initial part $\alpha \in [0, K(n))$ where the complexity is too small to specify the set $S_1 = \{0,1\}^n$, see the initial part of Section VI.

The little bumps in the sufficient statistic region $[K(K(x)), K(x)]$ in Figure 4 are due to the boundedness of the number of sufficient statistics. ◇

Example VI.3: **Sufficient Statistic:** Let us look at a coin toss example. Let k be a number in the range $0, 1, \ldots, n$ of complexity $\log n + O(1)$ given n and let x

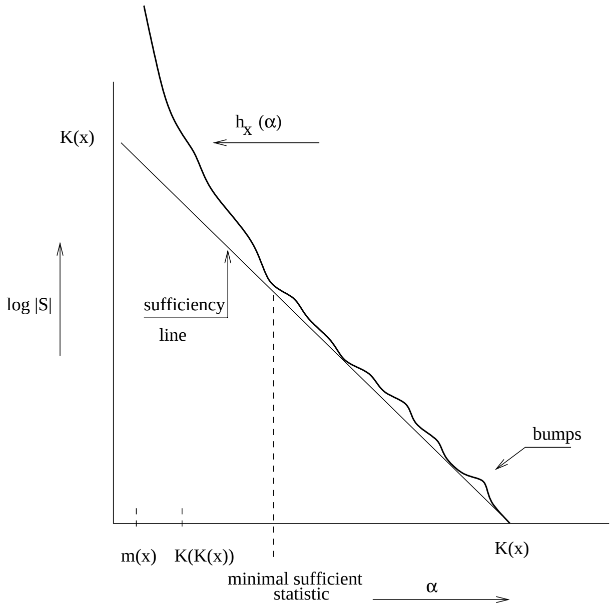

Fig. 4. Kolmogorov structure function

be a string of length n having k ones of complexity $K(x \mid n,k) \geq \log \binom{n}{k}$ given n, k. This x can be viewed as a typical result of tossing a coin with a bias about $p = k/n$. A two-part description of x is given by the number k of 1's in x first, followed by the index $j \leq \log|S|$ of x in the set S of strings of length n with k 1's. This set is optimal, since $K(x \mid n) = K(x, k \mid n) = K(k \mid n) + K(x \mid k, n) = K(S|n) + \log|S|$. ♢

Example VI.4: **Hierarchy of Sufficient Statistics:** Another possible application of the theory is to find a good summarization of the meaningful information in a given picture. All the information in the picture is described by a binary string x of length $n = ml$ as follows. Chop x into l substrings x_i $(1 \leq i \leq l)$ of equal length m each. Let k_i denote the number of ones in x_i. Each such substring metaphorically represents a patch of, say, color. The intended color, say "cobalt blue", is indicated by the number of ones in the substring. The actual color depicted may be typical cobalt blue or less typical cobalt blue. The smaller the randomness deficiency of substring x_i in the set of all strings of length m containing precisely k_i ones, the more typical x_i is, the better it achieves a typical cobalt blue color. The metaphorical "image" depicted by x is $\pi(x)$, defined as the string $k_1 k_2 \ldots k_l$ over the alphabet $\{0, 1, \ldots, m\}$, the set of colors available. We can now consider several statistics for x.

Let $X \subseteq \{0, 1, \ldots, m\}^l$ (the set of possible realizations of the target image), and let Y_i for $i = 0, 1, \ldots, m$ be a set of binary strings of length m with i ones (the set of realizations of target color i). Consider the set

$$S = \{x' : \pi(x') \in X, (x')_i \in Y_{k_i} \text{ for all } i = 1, \ldots, l\}$$

One possible application of these ideas are to gouge how good the picture is with respect to the given summarizing set S. Assume that $x \in S$. The set S is then a statistic for x that captures both the colors of the patches and the image, that is, the total picture. If S is a sufficient statistic of x then S perfectly expresses the meaning aimed for by the image and the true color aimed for in everyone of the color patches. Clearly, S summarizes the relevant information in x since it captures both image and coloring, that is, the total picture. But we can distinguish more sufficient statistics.

The set

$$S_1 = \{x' : \pi(x') \in X\}$$

is a statistic that captures only the image. It can be sufficient only if all colors used in the picture x are typical. The set

$$S_2 = \{x' : (x')_i \in Y_{k_i} \text{ for all } i = 1, \ldots, l\}$$

is a statistic that captures the color information in the picture. It can be sufficient only if the image is a random string of length l over the alphabet $\{0, 1, \ldots, m\}$, which is surely not the case for all the real images. Finally the set

$$A_i = \{x' : (x')_i \in Y_{k_i}\}$$

is a statistic that captures only the color of patch $(x')_i$ in the picture. It can be sufficient only if $K(i) \approx 0$ and all the other color applications and the image are typical. ♢

Example VI.5: **"Positive" and "Negative" Randomness:** (Continuing Example IV.20.) In [10] the existence of strings was shown for which essentially the singleton set consisting of the string itself is a minimal sufficient statistic. While a sufficient statistic of an object yields a two-part code that is as short as the shortest one part code, restricting the complexity of the allowed statistic may yield two-part codes that are considerably longer than the best one-part code (so the statistic is insufficient). This is what happens for the non-stochastic objects. In fact, for every object there is a complexity bound below which this happens—but if that bound is small (logarithmic) we call the object "stochastic" since it has a simple satisfactory explanation (sufficient statistic). Thus, Kolmogorov in [15] (full text given in Section I) makes the important distinction of an object being random in the "negative" sense by having this bound high (they have high complexity and are not typical elements of a low-complexity model), and an object being random in the "positive, probabilistic" sense by both having this bound small and itself having complexity considerably exceeding this bound (like a string x of length n with $K(x) \geq n$, being typical for the set $\{0,1\}^n$, or the uniform probability distribution over that set, while this set or probability distribution has complexity $K(n) + O(1) = O(\log n)$). We depict the distinction in Figure 5.

Corollary IV.9 establishes that for some constant C, for every length n, for every complexity $k \leq n$ and every $\alpha_0 \in [0, k]$, there are x's of length n and complexity $k \pm C \log n$ such that the minimal randomness deficiency $\beta_x(i) > n - k - C \log n$ for every $i \leq \alpha_0 - C \log n$ and $\beta_x(i) < C \log n$ for every $i \geq \alpha_0 + C \log n$. Fix $\varepsilon = C \log n$ and define

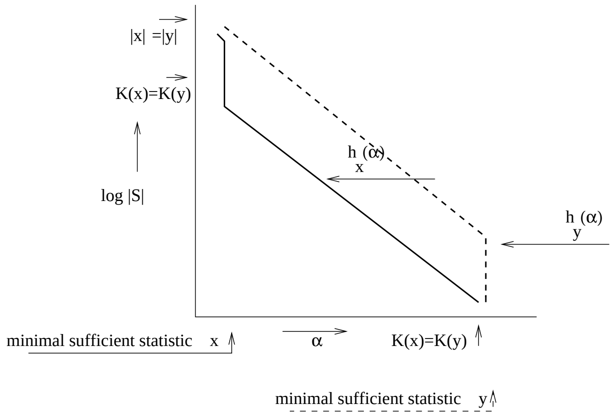

Fig. 5. Data string x is "positive random" or "stochastic" and data string y is "negative random" or "non-stochastic".

for all $s, t = 0, \dots, n/(2\varepsilon) - 1$ the set A_{st} of all n-length strings of complexity $K(x) \in [(2s-1)\varepsilon, (2s+1)\varepsilon)$ and such that the minimal randomness deficiency $\beta_x(i) > n - (2s+1)\varepsilon$ for every $i \leq (2t-1)\varepsilon$ and $\beta_x(i) < \varepsilon$ for every $i \geq (2t+1)\varepsilon$. Corollary IV.9 implies that every A_{st} is non-empty (let $\alpha_0 = 2t\varepsilon$, $k = 2s\varepsilon$). Note that A_{st} are pair-wise disjoint. Indeed, if $s \neq s'$ then A_{st} and $A_{s't'}$ are disjoint as the corresponding strings x, x' have different complexities. And if $t \neq t'$, say $t < t'$, then A_{st} and $A_{s't'}$ are disjoint, as the corresponding strings x, x' have different value of deficiency function in the point $i = (2t+1)\varepsilon$: $\beta_x((2t+1)\varepsilon) > n - (2s+1)\varepsilon \geq \varepsilon > \beta_{x'}((2t+1)\varepsilon)$.

Letting $k = \alpha_0 = n - \sqrt{n}$ we see that there are n-length non-stochastic strings of almost maximal complexity $n - \sqrt{n} \pm O(\log n)$ having significant $\sqrt{n} \pm O(\log n)$ randomness deficiency with respect to $\{0,1\}^n$ or, in fact, every other finite set of complexity less than $n - O(\log n)$! $\diamondsuit$

VII. Computability Questions

How difficult is it to compute the functions h_x, λ_x, β_x, and the minimal sufficient statistic? To express the properties appropriately we require the notion of functions that are not computable, but can be approximated monotonically by a computable function.

Definition VII.1: A function $f : \mathcal{N} \to \mathcal{R}$ is *upper semi-computable* if there is a Turing machine T computing a total function ϕ such that $\phi(x, t+1) \leq \phi(x, t)$ and $\lim_{t \to \infty} \phi(x, t) = f(x)$. This means that f can be computably approximated from above. If $-f$ is upper semi-computable, then f is lower semi-computable. A function is called *semi-computable* if it is either upper semi-computable or lower semi-computable. If f is both upper semi-computable and lower semi-computable, then we call f *computable* (or recursive if the domain is integer or rational).

Semi-computability gives no speed-of-convergence guaranties: even though the limit value is monotonically approximated we know at no stage in the process how close we are to the limit value. The functions $h_x(\alpha), \lambda_x(\alpha), \beta_x(\alpha)$ have finite domain for given x and hence can be given as a table—so formally speaking they are computable. But this evades the issue: there is no algorithm that computes these functions for given x and α. Considering them as two-argument functions we show the following (we actually quantify these):

- The functions $h_x(\alpha)$ and $\lambda_x(\alpha)$ are upper semi-computable but they are not computable up to any reasonable precision.
- Moreover, there is no algorithm that given x^* and α finds $h_x(\alpha)$ or $\lambda_x(\alpha)$.
- The function $\beta_x(\alpha)$ is not upper- or lower semi-computable, not even to any reasonable precision, but we can compute it given an oracle for the halting problem.
- There is no algorithm that given x and $K(x)$ finds a minimal sufficient statistic for x up to any reasonable precision.

Intuition: the functions h_x and λ_x (the ML-estimator and the MDL-estimator, respectively) can be monotonically approximated in the upper semi-computable sense. But the fitness function β_x cannot be monotonically approximated in that sense, nor in the lower semi-computable sense, in both cases not even up to any relevant precision.

The precise forms of these quite strong noncomputability and nonapproximability results are given in Appendix B.

VIII. Conclusion

When we compare statistical hypotheses S_0 and S_1 to explain x we should take into account three parameters: $K(S), K(x \mid S)$, and $\log |S|$. The first parameter is the *simplicity* of the theory S explaining the data. The difference $\delta(x|S) = \log |S| - K(x \mid S)$ (the randomness deficiency) shows *how typical* the data is with respect to S. The sum $\Lambda(S) = K(S) + \log |S|$ tells us how *short the two part code* of the data using theory S is, consisting of the code for S and a code for x simply using the worst-case number of bits possibly required to identify x in the enumeration of S. This second part consists of the full-length index ignoring savings in code length using possible non-typicality of x in S (like being the first element in the enumeration of S). We would like to define that S_0 is not worse than S_1 (as an explanation for x), in symbols: $S_0 \leq S_1$, if

- $K(S_0) \leq K(S_1)$;
- $\delta(x|S_0) \leq \delta(x|S_1)$; and
- $\Lambda(S_0) \leq \Lambda(S_1)$.

To be sure, this is not equivalent to saying that $K(S_0) \leq K(S_1), \delta(x|S_0) \leq \delta(x|S_1), \log |S_0| \leq \log |S_1|$. (The latter relation is stronger in that it implies $S_0 \leq S_1$ but not vice versa.) The algorithmic statistical properties of a data string x are fully represented by the set A_x of all triples $\langle K(S), \delta(x|S), \Lambda(S) \rangle$ such that $S \ni x$, together with a component wise order relation $\leq$ on the elements those triples. The complete characterization of how this set may look like (with $O(\log n)$-accuracy) is now known in the following sense.

Our results (Theorems IV.4, IV.8, IV.11) describe completely (with $O(\log n)$-accuracy) possible shapes of the closely related set B_x consisting of all triples $\langle \alpha, \beta, \lambda \rangle$ such that there is a set $S \ni x$ with $K(S) \leq \alpha$, $\delta(x \mid S) \leq \beta$, $\Lambda(S) \leq \lambda$. That is, $A_x \subseteq B_x$ and A_x and B_x have the same minimal triples. Hence, we can informally say that our results describe completely possible shapes of the set of triples $\langle K(S), \delta(x|S), \Lambda(S)\rangle$ for non-improvable hypotheses S explaining x. For example up to $O(\log n)$ accuracy, and denoting $k = K(x)$ and $n = |x|$:

(i) For every minimal triple (α, β, γ) in B_x we have $0 \leq \alpha \leq k$, $0 \leq \beta, \beta + k = \gamma \leq n$.

(ii) There is a triple of the form $(\alpha_0, 0, k)$ in B_x (the minimal such α_0 is the complexity of the minimal sufficient statistic for x). This property allows us to recover the complexity k of x from B_x.

(iii) There is a triple of the form $(0, \lambda_0 - k, \lambda_0)$ in B_x with $\lambda_0 \leq n$.

Previously, a limited characterization was obtained by V'yugin [26], [27] for the possible shapes of the projection of B_x on α, β-coordinates but only for the case when $\alpha = o(K(x))$. Our results describe possible shapes of the entire set B_x for the full domain of α (with $O(\log n)$-accuracy). Namely, let f be a non-increasing integer valued function such that $f(0) \leq n$, $f(i) = k$ for all $i \geq k$ and

$$\tilde{B}_f = \{\langle \alpha, \beta, \lambda\rangle \mid 0 \leq \alpha,\ f(\alpha) \leq \lambda,\ f(\alpha) - k \leq \beta\}.$$

For every x of length n and complexity k there is f such that

$$\tilde{B}_f + u \subset B_x \subset \tilde{B}_f - u \tag{VIII.1}$$

where $u = \langle c\log n, c\log n, c\log n\rangle$ for some universal constant c. Conversely, for every $k \leq n$ and every such f there is x of length n such that (VIII.1) holds for $u = \langle c\log K(n,f,k), c\log K(n,f,k), c\log K(n,f,k)\rangle$. Our results imply that the set B_x is not computable given x, k but is computable given x, k and α_0, the complexity of minimal sufficient statistic.

The major result of this work is that a finite set that witnesses $\lambda_x(\alpha)$ or $h_x(\alpha)$ (minimizes the log-cardinality or $\Lambda(S)$ of a set containing x of complexity at most α) simultaneously witnesses $\beta_x(\alpha)$ (minimizes the randomness deficiency of x with respect to a set containing it of complexity at most α). We have also addressed the non-computability of h_x (but it is upper semi-computable), and the fine structure of its shape (especially for α below the minimal sufficient statistic complexity).

Model determination deficiency: There is also the fourth important parameter, $K(S \mid x^*)$ reflecting the determinacy of hypothesis S by the data x. However, the equality $\log|S| + K(S) - K(x) = K(S \mid x^*) + \delta(x \mid S) + O(1)$ shows that this parameter can be expressed in α, β, h. The main result (III.1) establishes that $K(S \mid x^*)$ is logarithmic for *every* set S witnessing $h_x(\alpha)$. This also shows that there are at most polynomially many such sets.

Appendix

I. Proofs

Proof: **Lemma IV.2** The inequality $\lambda_x(\alpha) \leq h_x(\alpha) + \alpha$ is immediate. So it suffices to prove that $h_x(\alpha) + \alpha \leq \lambda_x(\alpha) + K(\alpha) + O(1)$. The proof of this inequality is based on the following:

Claim A.1: Ignoring additive $K(i)$ terms the function $h_x(i) + i$ does not increase:

$$h_x(i_2) + i_2 \leq h_x(i_1) + i_1 + K(i_2 \mid i_1) + O(1) \tag{A.1}$$

for $i_1 < i_2 \leq K(x)$.

Proof: Let S be a finite set containing x with $K(S) \leq i_1$ and $\log|S| = h_x(i_1)$. For every $m \leq \log|S|$, we can partition S into 2^m equal-size parts and select the part S' containing x. Then, $\log|S'| = \log|S| - m$ at the cost of increasing the complexity of S' to

$$K(S') \leq K(S) + m + K(m \mid K(S)) + O(1)$$

(we specify the part S' containing x by its index among all the parts). Choose

$$m = i_2 - K(S) - K(i_2 \mid K(S)) - c$$

for a constant c to be determined later. Note that

$$\begin{aligned} K(m \mid K(S)) &\leq K(i_2, K(i_2 \mid K(S)) \mid K(S)) + K(c) + c' \\ &= K(i_2 \mid K(S)) + K(c) + c'' \end{aligned}$$

for appropriate constants c', c''. The complexity of the resulting set S' is thus at most

$$\begin{aligned} K(S) + i_2 - K(S) - K(i_2 \mid K(S)) - c \\ + K(i_2 \mid K(S)) + K(c) + c'' \leq i_2, \end{aligned}$$

provided c is chosen large enough. Hence, $h_x(i_2) \leq \log|S'| = h_x(i_1) - m = h_x(i_1) - i_2 + K(S) + K(i_2 \mid K(S)) + c$, and it suffices to prove that $K(S) + K(i_2 \mid K(S)) \leq i_1 + K(i_2 \mid i_1) + O(1)$. This follows from the bound $K(i_2 \mid K(S)) \leq K(i_2 \mid i_1) + K(i_1 \mid K(S)) + O(1) \leq K(i_2 \mid i_1) + K(i_1 - K(S)) + O(1) \leq K(i_2 \mid i_1) + i_1 - K(S) + O(1)$. ∎

Let S witness $\lambda_x(\alpha)$. Substituting $K(S) = i_1$, $\alpha = i_2$ in (A.1) we obtain: $h_x(\alpha) + \alpha \leq h_x(K(S)) + K(S) + K(\alpha \mid K(S)) + O(1) \leq \Lambda(S) + K(\alpha) + O(1) = \lambda_x(\alpha) + K(\alpha) + O(1)$. ∎

Proof: **Theorem IV.4** (i) We first observe that for every x of length n we have $\lambda_x(K(n) + O(1)) \leq n + K(n) + O(1)$, as witnessed by $S = \{0,1\}^n$. At the other extreme, $\lambda_x(k + O(1)) = k + O(1)$, as witnessed by $S = \{x\}$.

Define $\lambda(i)$ by the equation $\lambda(i) - k = \max\{0, \lambda_x(i + K(n) + O(1)) - k - O(1)\}$. Then $\lambda_x = \mathcal{E}(\lambda)$ with $\varepsilon = \delta = K(n) + O(1)$, and λ satisfies the requirements of Item (i) of the theorem.

(ii) Fix $\lambda(i)$ satisfying the conditions in the theorem. It suffices to show that there is a string x of length n such that, for every $i \in [0, k]$, we have $\lambda_x(i) \geq \lambda(i)$ and $\lambda_x(i + \delta(i)) \leq \lambda(i) + \delta(i)$ for $\delta(i) = K(i, n, \lambda) + O(1)$. Then, with $\delta = \delta(k)$, we have $K(x) \leq \lambda_x(k + \delta) + O(1) \leq \lambda(k) + \delta + O(1) = k + \delta + O(1)$. And the inequality $\lambda_x(k) \geq \lambda(k) = k$ implies that $K(x) > k - O(1)$.

Claim A.2: For every length n, there is a string x of length n such that $\lambda_x(i) \geq \lambda(i)$ for every i in the domain of λ.

Proof: Fix a length n. If $\lambda_x(i) < \lambda(i)$ then x belongs to a set A with $\Lambda(A) < \lambda(i) \leq \lambda(0) \leq n$. The total number of elements in different such A's is less than $\sum_A 2^{n-K(A)} = 2^n \sum_A 2^{-K(A)} \leq 2^n$, where the second inequality follows by (II.2). ■

We prove Item (ii) by demonstrating that the lexicographically first x, as defined in Claim A.2, also satisfies $\lambda_x(i+\delta(i)) \leq \lambda(i)+\delta(i)$, for $\delta(i) = K(i,n,\lambda)+O(1)$ for all $i \in [0,k]$. It suffices to construct a set $S \ni x$ of cardinality $2^{\lambda(i)-i}$ and of complexity at most $i+\delta(i)$, for every $i \in [0,k]$.

For every fixed $i \in [0,k]$ we can run the following:

Algorithm: Let A be a set variable initially containing all strings of length n, and let S be a set variable initially containing the $2^{\lambda(i)-i}$ first strings of A in lexicographical order. Run all programs of length at most n dovetail style. Every time a program p of some length j halts, $\lambda(j)$ is defined, and p prints a set B of cardinality at most $2^{\lambda(j)-j}$, we remove all the elements of B from A (but not from S); we call a step at which this happens a *j-step.* Every time $S \cap A$ becomes empty at a j-step, we replace the contents of S by the set of the $2^{\lambda(i)-i}$ first strings in lexicographical order of (the current contents of) A. Possibly, the last replacement of S is incomplete because there are less than $2^{\lambda(i)-i}$ elements left in A. It is easy to see that $x \in S \setminus A$ just after the final replacement, and stays there forever after, even though some programs in the dovetailing process may still be running and elements from A may still be eliminated.

Claim A.3: The contents of the set S is replaced at most 2^{i+1} times.

Proof: There are two types of replacements that will be treated separately.

Case 1: Replacement of the current contents of S where at some j-step with $j \leq i$ *at least one* element was removed from the current contents $S \cap A$. Trivially, the number of this type of replacements is bounded by the number of j-steps with $j < i$, and hence by the number of programs of length less than i, that is, by 2^i.

Case 2: Replacement of the current contents of S where *every one* of the $2^{\lambda(i)-i}$ elements of the current contents of S is removed from A by j-steps with $j \geq i$. Let us estimate the number of this type of replacements: Every element x removed at a j-step with $j \geq i$ belongs to a set B with $\Lambda(B) \leq \lambda(j) \leq \lambda(i)$. The overall cumulative number of elements removed from A on j-steps with $j \geq i$ is bounded by $\sum_B 2^{\lambda(i)-K(B)} \leq 2^{\lambda(i)}$, where the inequality follows by (II.2). Hence replacements of the second type can happen at most $2^{\lambda(i)-(\lambda(i)-i)} = 2^i$ times. ■

By Claim A.3, S stabilizes after a certain number of j-steps. That number may be large. However, the number of replacements of S is small. The final set $S \ni x$ has cardinality $2^{\lambda(i)-i}$, and can be specified by the number of replacements resulting in its current contents (as in Claim A.3), and by i, n, λ. This shows that $K(S) \leq i + K(i,n,\lambda) + O(1)$. ■

Proof: **Theorem IV.8** The statement of the theorem easily follows from the following two inequalities that are valid for every x (where $n = |x|$ and $k = K(x)$):

$$\beta_x(i) + k \leq \lambda_x(i) + O(1), \quad \text{for every } i \leq k; \text{ and} \tag{A.2}$$
$$\lambda_x(i + O(\log n)) \leq \beta_x(i) + k + O(\log n), \tag{A.3}$$
$$\text{for every } i \text{ satisfying } K(n) + O(1) \leq i \leq k.$$

It is convenient to rewrite the formula defining $\delta(x \mid A)$ using the symmetry of information (II.3) as follows:

$$\begin{aligned} \delta(x \mid A) &= \log|A| + K(A) - K(A \mid x^*) - k + O(1) \\ &= \Lambda(A) - K(A \mid x^*) - k + O(1). \end{aligned} \tag{A.4}$$

Ad (A.2): This is easy, because for every set $S \ni x$ witnessing $\lambda_x(i)$ we have $\delta(x \mid S) \leq \Lambda(S) - k + O(1) = \lambda_x(i) - k + O(1)$ and $\beta_x(i) \leq \delta(x \mid S)$.

Ad (A.3): This is more difficult. By (A.4), and the obvious $K(A \mid x^*) \leq K(A \mid x) + O(1)$, it suffices to prove that for every $A \ni x$ there is an $S \ni x$ with

$$K(S) \leq K(A) + O(\log m),$$
$$\log|S| \leq \log|A| - K(A \mid x) + O(\log m),$$

where $m = \Lambda(A)$. Indeed for every A witnessing $\beta_x(i)$ the set S will witness $\lambda_x(i + O(\log n)) \leq \beta_x(i) + k + O(\log n)$ (note that $m = \log|A| + K(A) = K(x \mid A^*) + \beta_x(i) + K(A) \leq 3n + O(\log n)$ provided $i \geq K(n) + O(1)$). The above assertion is only a little bit easier to prove than the one in Lemma A.4 below that also suffices. Since we need this lemma in any case in the proof of Theorem IV.11 we state and prove it right now.

Lemma A.4: For every $A \ni x$ there is $S \ni x$ with $K(S) \leq K(A) - K(A \mid x) + O(\log m)$ and $\lceil \log|S| \rceil = \lceil \log|A| \rceil$ (where $m = \Lambda(A)$).

Proof: Fix some $A_0 \ni x$ and let $m = \Lambda(A_0)$. Our task is the following: Given $K(A_0), \lceil \log|A_0| \rceil, K(A_0 \mid x)$, to enumerate a family of at most $2^{K(A_0)-K(A_0|x)+O(\log m)}$ different sets S with $\log|S| = \lceil \log|A_0| \rceil$ that cover all y's covered by sets A, with $K(A) = K(A_0)$, $K(A \mid y) = K(A_0 \mid x)$ and $\lceil \log|A| \rceil = \lceil \log|A_0| \rceil$. Since the complexity of each enumerated S does not exceed $K(A_0) - K(A_0 \mid x) + O(\log m) + K(K(A_0), \lceil \log|A_0| \rceil, K(A_0 \mid x)) + O(1) = K(A_0) - K(A_0 \mid x) + O(\log m)$ the lemma will be proved. The proof is by running the following:

Algorithm: Given $K(A_0), \lceil \log|A_0| \rceil, K(A_0 \mid x)$ we run all programs dovetail style. We maintain auxiliary set-variables C, U, D, all of them initially $\emptyset$. Every time a new program p of length $K(A_0)$ in the dovetailing process halts, with as output a set A with $\lceil \log|A| \rceil = \lceil \log|A_0| \rceil$, we execute the following steps:

Step 1: Update $U := U \cup A$.

Step 2: Update $D := \{y \in U \setminus C\colon y$ is covered by at least $t = 2^{K(A_0|x)-\delta}$ different generated A's$\}$, where $\delta = O(\log m)$ will be defined later.

Step 3: This step is executed only if there is $y \in D$ that is covered by at least $2t$ different generated A's. Enumerate as much new disjoint sets S as are needed to cover D: we just chop D into parts of size $2^{\lceil \log|A_0| \rceil}$ (the last part may be incomplete) and name those parts the new sets S. Every time a new set S is enumerated, update $C := C \cup S$.

Claim A.5: The string x is an element of some enumerated S, and the number of enumerated S's is at most $2^{K(A_0)-K(A_0|x)+O(\log m)}$.

Proof: By way of contradiction, assume that x is not an element of the enumerated S's. Then there are less than $2^{K(A_0|x)-\delta+1}$ different generated sets A such that $x \in A$. Every such A therefore satisfies $K(A \mid x) \leq K(A_0 \mid x) - \delta + O(\log m) < K(A_0 \mid x)$ if δ is chosen appropriately. Since A_0 was certainly generated this is a contradiction.

It remains to show that we enumerated at most $2^{K(A_0)-K(A_0|x)+O(\log m)}$ different S's. Step 3 is executed only once per t executions of Step 1, and Step 1 is executed at most $2^{K(A_0)}$ times. Therefore Step 3 is executed at most $2^{K(A_0)}/t = 2^{K(A_0)-K(A_0|x)+\delta}$ times. The number of S's formed from incomplete parts of D's in Step 3 is thus at most $2^{K(A_0)-K(A_0)+\delta}$. Let us bound the number of S's formed from complete parts of D's. The total number of elements in different A's generated is at most $2^{K(A_0)+\lceil \log |A_0| \rceil}$ counting multiplicity. Therefore the number of elements in their union, having multiplicity $2^{K(A_0|x)-\delta}$ or more, is at most $2^{K(A_0)+\lceil \log |A_0| \rceil - K(A_0|x)+\delta}$. Every S formed from a complete part of a set D in Step 3 accounts for $2^{\lceil \log |A_0| \rceil}$ of them. Hence the number of S's formed from complete parts of D's is at most $2^{K(A_0)-K(A_0|x)+\delta}$. ■

■

■

Proof: **Theorem IV.11** By Lemma A.4 there is $S \ni x$ with $K(S) \leq K(A) - K(A \mid x) + O(\log \Lambda(A))$ and $\lceil \log |S| \rceil = \lceil \log |A| \rceil$.

Let us upper bound first $K(S)$. We have

$$\begin{aligned} K(S) &\leq K(A) - K(A \mid x) = \delta(x|A) + k - \log |A| \\ &= \beta_x(\alpha) + k - \log |A| + (\delta(x|A) - \beta_x(\alpha)) \\ &\leq \lambda_x(\alpha) - \log |A| + (\delta(x|A) - \beta_x(\alpha)). \end{aligned}$$

(all inequalities are valid up to $O(\log \Lambda(A))$ additive term). The obtained upper bound is obviosly equivalent to the first upper bound of $K(S)$ in the theorem. As $\log |S| = \log |A|$ it gives the upper bound of $\Lambda(S)$ from the theorem. Finally, as $\lambda_x(\alpha) \leq h_x(\alpha) + \alpha + O(1)$ we obtain $K(S) \leq \alpha + (h_x(\alpha) - \log |A|) + (\delta(x|A) - \beta_x(\alpha))$ (up to $O(\log \Lambda(A))$ additive term). ■

Proof: **Theorem V.2** We first show that $|m_x^l| \leq K(x) + O(\log l)$ for every x with $K(x) \leq l$. Indeed, given $x, l, |m_x^l|$ and the $|N^l| - |m_x^l|$ least significant bits of N^l we can find N^l: find I_x^l by enumerating D until a pair $\langle x, i \rangle$ with $i \leq l$ appears and then complete m_x^l by using the $|m_x^l|$ most significant bits of the binary representation of I_x^l. Given l and N^l we can find, using a constant-length program, the lexicographically first string not in N^l. By construction, this string has complexity at least $l+1$. Then, $l \leq K(N^l) + O(\log l) \leq K(x) + |N^l| - |m_x^l| + O(\log l) \leq K(x) + l - |m_x^l| + O(\log l)$ (use $|N^l| \leq l + O(1)$). Thus, $|m_x^l| \leq K(x) + O(\log l)$.

Let x be the string of complexity at most i with maximum I_x^l. Given m_x^l and $i, l, |N^l|$ we can find all strings of complexity at most i by enumerating D until N pairs $\langle y, j \rangle$ with $j \leq l$ appear, where N is the number whose binary representation has prefix $m_x^l 1$ and then $(|N^l| - |m_x^l| - 1)$ zeros. Since $|m_x^l| \leq i + O(\log l)$, this proves $K(N^i \mid N_i^l) = O(\log l)$. Since $K(N^i) \geq i - O(\log i) \geq K(N_i^l) - O(\log i)$ we have $K(N_i^l \mid N^i) = O(\log l)$. ■

Proof: **Theorem V.4** (i) If the $(i+1)$st most significant bit of N^l is "1," then all the numbers with binary representation of the form $N_i^l 0 * * \cdots *$ are used as indexes of some y with $K(y) \leq l$, that is, S_i^l has exactly $2^{|N^l|-i-1}$ elements. We can find S_i^l given l, i, $|N^l|$ and N_i^l by enumerating all its elements. On the other hand, N_i^l can be found given S_i^l and i, l as the first i bits of I_x^l for every $x \in S_i^l$.

(ii) Since $i = |m_x^l|$, the largest common prefix of binary representation of I_x^l and N^l has the form $N_i^l 0 * * \cdots *$ and the $(i+1)$st most significant bit of N^l is 1. In particular, $x \in S_i^l$.

Let $J = \max\{I_y^l \mid y \in S\}$. As $x \in S$, we have $J \geq I_x^l$. We can find N_i^l given i, l and S by finding J and taking the i first bits of J. Given N_i^l we can find S_i^l. Hence $K(S_i^l \mid S) = O(\log l)$. Therefore $K(S_i^l) \leq K(S) + O(\log l)$. By Item (i) and by previous theorem we have $K(S_i^l) = i + O(\log l)$. Again by Item (i) we have $\Lambda(S_i^l) \leq l + O(\log l) = \Lambda(S) + O(\log l)$.

(iii) Let $i = |m_x^l|$. We distinguish two cases.

Case 1: $i \geq \alpha$. Then $K(N^\alpha \mid S) \leq K(N^\alpha \mid S_i^l) + O(\log l) \leq K(N^\alpha \mid N^i) + O(\log l) = O(\log l)$. And $K(S \mid N^\alpha) = K(S) - K(N^\alpha) + O(\log l) = K(S) - \alpha + O(\log l)$.

Case 2: $i < \alpha$. Let $A = S_i^l$. As $\Lambda(S_i^l) \leq \Lambda(S) + O(\log l)$ we need to prove that $K(S_i^l) \leq \alpha - K(N^\alpha|S)$ and $K(S_i^l) \leq K(S) - K(S|N^\alpha)$ up to $O(\log l)$ additive term. We have

$$\begin{aligned} K(S_i^l) &= K(N^\alpha) - K(N^\alpha|S_i^l) + O(\log l) \\ &\leq \alpha - K(N^\alpha|S) + O(\log l) \end{aligned}$$

and

$$\begin{aligned} K(S_i^l) &= K(S) - K(S|S_i^l) + O(\log l) \\ &\leq K(S) - K(S|N^\alpha) + O(\log l). \end{aligned}$$

■

II. Computability properties

Structure Function: It is easy to see that $h_x(\alpha)$ or $\lambda_x(\alpha)$, and the finite set that witnesses its value, are upper semi-computable: run all programs of length up to α dovetailed fashion, check whether a halting program produced a finite set containing x, and replace the previous candidate with the new set if it is smaller.

The next question is: Is the function $\lambda_x(\alpha)$, as the function of two arguments, computable? Of course not, because if this were the case, then we could find, given every large k, a string of complexity at least k. Indeed, we know that there is a string x for which $\lambda_x(k) > k$. Applying the algorithm to all strings in the lexicographical order find the first such x. Obviously $K(x) \geq k - O(1)$. But it is known that we cannot prove that $K(x) > k$ for sufficiently large k, [13].

Assume now that we are given also $K(x)$. The above argument does not work any more but the statement remains true: $\lambda_x(\alpha)$ is not computable even if the algorithm is given $K(x)$.

Assume first that the algorithm is required to output the correct answer given any approximation to $K(x)$. We show that no algorithm can find λ that is close to $\lambda_x(\alpha)$ for some $0 \ll \alpha \ll K(x)$.

Theorem B.1: For every constant c there is a constant d such the following holds. There is no algorithm that for infinitely many k, given k and x of length $k + d \log k$ with $|K(x) - k| \leq 2\log k$, always finds λ such that there is $2\log k \leq \alpha \leq k$ with $|\lambda_x(\alpha) - \lambda| < c \log k$.

Proof: Fix c. The value of d will be chosen later. The proof is by contradiction. Let A be some algorithm. We want to fool it on some pair $\langle x, k\rangle$.

Fix large k. We will construct a set S of cardinality $2^{k-2\log k}$ such that every string x in S has length $k+d\log k$ and complexity at most $k+2\log k$, and the algorithm halts on $\langle x, k\rangle$ and outputs $\lambda \geq (c+1)\log k$. This is a contradiction. Indeed, there is $x \in S$ with $K(x) \geq k - 2\log k$. Hence the output λ of A on $\langle x, k\rangle$ is correct, that is, there is α with $2\log k \leq \alpha \leq k$ and $|\lambda_x(\alpha) - \lambda| < c\log k$. Then $\lambda_x(\alpha) > \log k$. On the other hand, $\lambda(2\log k) \leq k$ as witnessed by S. Thus we obtain

$$k < \lambda_x(\alpha) \leq \lambda_x(2\log k) \leq k,$$

a contradiction.

Run in a dovetailed fashion all programs of length k or less. Start with x equal to the first string of length $k + d\log k$ and with $S = B = \emptyset$. Run A on $\langle x, k\rangle$ and include in B all strings x' such that either a program p of length at most k has halted and output a set $C \ni x'$ with $|p| + \log|C| \leq k + (2c+1)\log k$, or we find out that $K(x') < k - 2\log k$. Once x gets in B we change x to the first string of length $k + d\log k$ outside $B \cup S$. (We will show that at every step it holds $|B \cup S| < 2^{k+d\log k}$.)

We proceed in this way until $A(x, k)$ prints a number λ or the number of changes of x exceed 2^{k+2}. (Actually, we will prove that the number of changes of x does not exceed $2^{k+1} + 2^{k-2\log k}$.) Therefore $K(x) \leq k + 2\log k$ for all our x's so we eventually will find x such that $A(x, k)$ outputs a result λ. If $\lambda \geq k + (c+1)\log k$ then include x in S and then change x to the first string of length $k + d\log k$ outside (the current version of) $B \cup S$. Otherwise, when $\lambda < k + (c+1)\log k$, let $\tilde{\lambda}_x$ be the current approximation of λ_x. We know that x is outside all known sets C with $K(C) \leq k, K(C)+\log|C| \leq k+(2c+1)\log k$. Therefore, for every $\alpha \leq k$ it holds $\tilde{\lambda}_x(\alpha) > k + (2c+1)\log k$ and hence $|\tilde{\lambda}_x(\alpha) - \lambda| > c\log k$. This implies that either $K(x) < k - 2\log k$ or $\tilde{\lambda}_x$ differs from λ_x. So we are sure that at least one more program of length k or less still has to halt. We wait until this happens, then include x in B and change x to the first string of length $k + d\log k$ outside $B \cup S$.

Once we get $2^{k-2\log k}$ elements in S we halt. Every change of x is caused by a halting of a new program of length at most k or by including x in S, thus the total number of changes does not exceed $2^{k+1} + 2^{k-2\log k}$.

Note that at every step we have

$$|B \cup S| \leq 2^{k-2\log k} + 2^{k+(2c+1)\log k} + 2^{k-2\log k} < 2^{k+d\log k}$$

provided that $d > 2c + 1$. ■

What if the algorithm is required to approximate λ_x only if it is given the precise value of $K(x)$? We are able to prove that in this case the algorithm cannot compute $\lambda_x(\alpha)$ too. It is even impossible to approximate the complexity of minimal sufficient statistic. To formulate this result precisely consider the following promise problem:

Input: $x, k = K(x), \alpha \in [\varepsilon, k - \varepsilon]$.

Output:

1, if $\lambda_x(\alpha - \varepsilon) < k + 6\log k$,

0, if $\lambda_x(\alpha + \varepsilon) > k + 3\varepsilon$.

If neither of two above cases occurs the algorithm may output any value or no value at all.

Theorem B.2: There is no algorithm A solving this promise problem for all x and $\varepsilon = |x|/10\log|x|$.

Corollary B.3: There is no algorithm that given $x, k = K(x)$ finds an integer valued function λ on $[0, k]$ such that $\lambda_x = \mathcal{E}(\lambda)$ for $\varepsilon = \delta = |x|/10\log|x|$.

Indeed, if there were such algorithm we could solve the above promise problem by answering 1 when $\lambda(\alpha) \leq k+2\varepsilon$ and 0 otherwise.

Proof: The proof is by contradiction. The idea is as follows. Fix large k. We consider $N = O(\log k)$ points $\alpha_1, \dots, \alpha_N$ that divide the segment $[0, k]$ into equal parts. We lower semicompute λ_x and $K(x)$ for different x's of length about $k + 4\varepsilon$. We are interested in strings x with $\tilde{\lambda}_x(\alpha_1 + \varepsilon) > k + 3\varepsilon$ where $\tilde{\lambda}_x$ is the current approximation to λ_x. By counting arguments there are many such strings. We apply the algorithm to $\langle x, \tilde{K}(x), \alpha_1\rangle$ for those x's, where $\tilde{K}(x)$ stands for the currently known upper bound for $K(x)$. Assume that $A(x, \tilde{K}(x), \alpha_1)$ halts. If the answer is 1 then we know that $K(x) < \tilde{K}(x)$ or $\lambda_x(\alpha_1 + \varepsilon) < \tilde{\lambda}_x(\alpha_1 + \varepsilon)$ and we continue lower semicomputation until we get know which of two values $\tilde{K}(x)$ or $\tilde{\lambda}_x(\alpha_1 + \varepsilon)$ gets smaller. If the latter is decreased we just remove x (the total number of removed x will not increase $2^{k+3\varepsilon}$ and thus they form a small fraction of strings of length $k + 4\varepsilon$). If for many x's the answer is 0 we make those answers incorrect by including those x's in a set of cardinality $2^{k-\alpha_1+2\varepsilon}$ and complexity $\alpha_1 - \varepsilon$. Then for all such x's $\lambda_x(\alpha_1 - \varepsilon) < k + \varepsilon$ and thus algorithm's answer is incorrect. Hence $K(x) < \tilde{K}(x)$ and we continue lower semicomputation. For all those x's for which $\tilde{K}(x)$ is decreased we repeat the trick with α_2 in place of α_1. In this way we will force $\tilde{K}(x)$ to decrease very fast for many x's. For most of x's $\tilde{K}(x)$ will become much less than k, which is impossible.

Here is the detailed construction. Fix large k. Let $N = 3\log k$, $\delta = k/9\log k$ (one third of the distance between consecutive α_i), $\alpha_i = k - 3\delta i + \delta$, $n = k + 4\delta$ (the length of x). The value of parameter ε is chosen to be slightly less than δ (we will need that $\delta > \varepsilon + 4\log k$ for large enough k).

We will run all the programs of length at most $k' = k + 2\log k$ and the algorithm A on all possible inputs in a dovetailed fashion.

We will define a set X of 2^k strings of length n. Our action will be determined by k only, hence $K(x) \le k + 2\log k = k'$ for every $x \in X$ provided k is large enough. We will also define some small sets B_l for $l = 1, \dots, N$, the sets of "bad" strings and B will denote their union. Every B_l will have at most $2^{k-\delta}$ elements. We start with $B_l = \emptyset$ for $l = 1, \dots, N$.

We make 2^k stages. At every stage consider the sets

$$X_l = \{x \in X \setminus B \colon \tilde{K}(x) = k' + 1 - l\} \quad \text{for } l = 1, \dots, N,$$
$$X_0 = \{x \in X \setminus B \colon \tilde{K}(x) > k'\}.$$

Before and after every stage the following invariant will be true.

(1) $|X_l| < 2^{3\delta l}$ for every $0 \le l \le N$; in particular $X_0 = \emptyset$.
(2) For all $1 \le l \le N$ for all $x \in X_l$ it holds $A(x, \tilde{K}(x), \alpha_l) = 0$.
(3) For all $0 \le l < i \le N$ and all $x \in X_l$ it holds $\tilde{\lambda}_x(\alpha_i + \delta) > k + 3\delta$.
(4) $|B_i| \le 2^{3i\delta - 3\delta + 1} \times$(the number of programs of length at most $k - 3i\delta + 2\delta$ that have halted so far).

At the start all X_l's and B_i's are empty so the invariant is true. Each stage starts by including a new element in X. This element is the first string x_0 of length $n = k + 4\delta$ outside X such that $\tilde{\lambda}_x(\alpha) > k + 3\delta$ for all $\alpha \le k$. Thus by the choice of x_0 the assertions (3) and (4) remain true but (1) and (2) may not.

We claim that continuing the dovetailing and updating properly B_i's we eventually make every one of (1), (2), (3) and (4) true. During the dovetailing the sets X_l change (an element can move from X_l to X_i for $i > l$ and even to $X \setminus (X_0 \cup \dots \cup X_N)$). We will denote by $\tilde{X}_l$ the version of X_l at the beginning of the stage (and $\tilde{X}_0 = \{x_0\}$) and keep the notations X_l, B_i for current versions of X_l, B_i, respectively. The rule to update B_i's is very simple: once at some step of the dovetailing a new set C of complexity at most $k - 3i\delta + 2\delta = \alpha_i + \delta$ appears, we include in B_i all the elements of the set $\bigcup_{j=0}^{i-1} \tilde{X}_j$. As $X_l \subset \bigcup_{j=0}^{l} \tilde{X}_j$ this keeps (3) true. Moreover, this keeps true also the following assertion:

(5) For all $1 \le l \le N$ for all $x \in X_l \setminus \tilde{X}_l$ it holds $\tilde{\lambda}_x(\alpha_l + \delta) > k + 3\delta$.

And this also keeps (4) true since $\bigcup_{j=0}^{i-1} |\tilde{X}_j| < 1 + \sum_{j=1}^{i-1} 2^{3\delta j} < 2^{3\delta(i-1)+1}$.

We continue the dovetailing and update B_i's as described until both (1) and (2) are true. Let us prove that this happens eventually. It suffices to show that if (3), (4) and (5) are true but (2) is not, or (2), (3), (4) and (5) are true but (1) is not then at least one program of length $\le k'$ will halt or $A(x, \tilde{K}(x), \alpha_l)$ is undefined for some l and some $x \in X_l$.

Consider the second case: (2), (3), (4) and (5) are true but (1) is not. Pick l such that $|X_l| \ge 2^{3\delta l}$. If $l = 0$, that is, $\tilde{K}(x_0) > k'$, we are done, as $K(x_0) \le k'$. Otherwise, let S consist of the first $2^{3\delta l}$ elements in X_l. We claim that $K(S) \le k - 3l\delta + 4\log k \le \alpha_l - \varepsilon$. To prove the claim we will show that all $S \subset X_l$ obtained in this way are pairwise disjoint, therefore their number is at most $2^k / 2^{3l\delta}$. Thus S may be identified by k, l and its index among all such $S \subset X_l$.

Therefore for all $x \in S$ we have $\lambda_x(\alpha_l - \varepsilon) < k + 4\log k < \tilde{K}(x) + 6\log \tilde{K}(x)$ and the value $A(x, \tilde{K}(x), \alpha_l) = 0$ is not correct. This implies that $\tilde{K}(x)$ is not correct for all $x \in S$. We continue the dovetailing until all elements of S move outside X_l. Then S becomes disjoint with $X_0 \cup \dots \cup X_l$ and therefore it will be disjoint with all future versions of X_l.

Consider the first case: (3), (4) and (5) are true but (2) is not. Pick l and $x \in X_l$ such that $A(x, \tilde{K}(x), \alpha_l)$ is undefined or $A(x, \tilde{K}(x), \alpha_l) = 1$. If $A(x, \tilde{K}(x), \alpha_l)$ is undefined then we are done: since $\tilde{\lambda}_x(\alpha_l + \varepsilon) \ge \tilde{\lambda}_x(\alpha_l + \delta) > k + 3\delta > \tilde{K}(x) + 3\varepsilon$, either $\tilde{\lambda}_x$ or $\tilde{K}$ will decrease, or $A(x, \tilde{K}(x), \alpha_l)$ will get defined. Consider the other case. Obviously $x \notin X_l \setminus \tilde{X}_l$. By (5) we have $\tilde{\lambda}_x(\alpha_l + \varepsilon) \ge \tilde{\lambda}_x(\alpha_l + \delta) > k + 3\delta \ge \tilde{K}(x) + 3\varepsilon$. Therefore $\lambda_x(\alpha_l + \delta) < \tilde{\lambda}_x(\alpha_l + \delta)$ or $\tilde{K}(x) < K(x)$ and we are done.

After 2^k stages the set $|X|$ has 2^k elements and we have a contradiction. Indeed, all $X_1, \dots, X_N$ form a very small part of X because of (1). The sets $B_1, \dots, B_N$ together form also a very small part of X because of (4). Thus for most strings $x \in X$ it holds $\tilde{K}(x) < k' - N + 1 \ll k$ which is a contradiction. ■

Remark B.4: Let us replace in the above promise problem $K(x)$, the prefix complexity of x, by $C(x)$, the plain complexity of x. For the modified problem we can strengthen the above theorem by allowing $\varepsilon = |x|/c$ where the constant c depends on the reference computer. Indeed for every $x \in X$ we have $C(x) \le k + O(1)$: every $x \in X$ can be described by its index in X in exactly k bits and the value of k may be retrieved from the length of the description of x. Therefore we will need $N = O(1)$ to obtain a contradiction. ♢

After a discussion of these results, Andrei A. Muchnik suggested, and proved, that if we are also given an α_0 such that $\lambda_x(\alpha_0) \approx K(x)$ but $\lambda_x(\alpha)$ is much bigger than $K(x)$ for α much less than α_0 (which is therefore the complexity of the minimal sufficient statistic), then we can compute λ_x over all of its domain. This result underlines the significance of the information contained in the *minimal* sufficient statistic:

Theorem B.5: There are a constant $c \ge 0$ and an algorithm that given any x, k, α_0 with $K(x) \le k \le \lambda_x(\alpha_0)$ finds a non-increasing function λ defined on $[0, k]$ such that $\lambda_x = \mathcal{E}(\lambda)$ with $\delta = \lambda_x(\alpha_0) - K(x) + O(1)$ and $\varepsilon = \alpha_0 - \alpha_1 + c\log k$ where $\alpha_1 = \min\{\alpha : \lambda_x(\alpha) \le k + c\log k\}$.

Proof: The algorithm is a follows. Let $D_k = \{\langle y, i\rangle \mid K(y) \le i \le k\} \subset D$. Enumerate pairs $\langle y, i\rangle \in D_k$ until a pair $\langle x, i_0\rangle$ appears and form a list of all enumerated pairs. For $\alpha \le \alpha_0$ define $\lambda(\alpha)$ to be the minimum $i + \log|S|$ over all $S \ni x$ such that a pair $\langle x, i\rangle$ with $i \le \alpha$ is in the list. For $\alpha_0 < \alpha \le k$ let $\lambda(\alpha) = k$.

For every $\alpha > \alpha_0$ we have $\lambda_x(\alpha) \ge K(x) - O(1) \ge k - \lambda_x(\alpha_0) + K(x) - O(1) = \lambda(\alpha) - \delta$ and $\lambda_x(\alpha) \le \lambda_x(\alpha_0) \le k + (\lambda_x(\alpha_0) - K(x)) \le \lambda(\alpha) + \delta$.

For every $\alpha \le \alpha_0$ we have $\lambda_x(\alpha) \le \lambda(\alpha)$. So it remains to show that for every $\varepsilon \leq \alpha \leq \alpha_0$ we have $\lambda(\alpha) \leq \lambda_x(\alpha - \varepsilon) + \delta$. We will prove a stronger statement: $\lambda(\alpha) = \lambda_x(\alpha)$ for every $\alpha \leq \alpha_0 - \varepsilon$ provided c is chosen appropriately. To prove this it suffices to show that all for all S with $K(S) \leq \alpha_0 - \varepsilon$ the pair $\langle S, K(S) \rangle$ belongs to the list.

By Theorem V.4 Item (i) we have $\lambda_x(|m_x^k| + c_1 \log k) \leq k + c_2 \log k$. That is, $\alpha_1 \leq |m_x^k| + c_1 \log k$ if $c \geq c_2$ and $\alpha_0 - \varepsilon = \alpha_0 - \alpha_0 + \alpha_1 - c \log k \leq |m_x^k| + (c_1 - c) \log k$.

From the proof of Theorem V.2 we see that there is a constant c_3 such that for every y with $K(y) \leq |m_x^k| - c_3 \log k$ the index of $\langle y, K(y) \rangle$ in the enumeration of D_k has less than $|m_x^k|$ common bits with N^k. Assuming that $c \geq c_1 + c_3$ we obtain that the indexes of all pairs $\langle y, K(y) \rangle$ with $K(y) \leq \alpha_0 - \varepsilon$ in the enumeration of D_k are less than I_x^k. ■

Randomness Deficiency Function: The function $\beta_x(\alpha)$ is computable from x, α given an oracle for the halting problem: run all programs of length $\leq \alpha$ dovetailed fashion and find all finite sets S containing x that are produced. With respect to all these sets determine the conditional complexity $K(x \mid S^*)$ and hence the randomness deficiency $\delta(x \mid S)$. Taking the minimum we find $\beta_x(\alpha)$. All these things are possible using information from the halting problem to determine whether a given program will terminate or not. It is also the case that the function $\beta_x(\alpha)$ is upper semi-computable from $x, \alpha, K(x)$ up to a logarithmic error: this follows from the semi-computability of $\lambda_x(\alpha)$ and Theorem IV.8. More subtle is that β_x is not semi-computable, not even within a large margin of error:

Theorem B.6: The function $\beta_x(\alpha)$ is

(i) not lower semi-computable to within precision $|x|/3$; and

(ii) not upper semi-computable to within precision $|x|/\log^4 |x|$.

Proof: (i) The proof is by contradiction. Assume Item (i) is false. Choose an arbitrary length n. Let β be a function defined by $\beta(i) = \frac{n}{2}$ for $0 \leq i \leq \frac{n}{3}$, and equal 0 otherwise. Then the function β_x with x of length n, corresponding to β, by Corollary IV.9, has x with $k = K(x)$ satisfying $\beta(0) = n - k \pm O(\log n)$ so that $k = \frac{n}{2} \pm O(\log n)$. Moreover, $\beta_x(i) = \frac{n}{2} \pm O(\log n)$ for $O(\log n) < i \leq \frac{n}{3} - O(\log n)$, and $\beta_x(i) = O(\log n)$ for $i > \frac{n}{3} + O(\log n)$. Write the set of such x's as X. By dovetailing the lower approximation of $\beta_x(i)$ for all x of length n and some i with $\frac{n}{8} \leq i \leq \frac{n}{4}$, by assumption on lower semi-computability of β_x, we must eventually find an x, if not $x \notin X$ then $x \in X$, for which the lower semi-computation of $\beta_x(i)$ exceeds $\frac{n}{2} - \frac{n}{3} - O(\log n)$. But we know from Corollary IV.9 that $\beta_x(i) = O(\log n)$ for $i > K(x) + O(\log n)$, and hence we have determined that $i - O(\log n) < K(x)$. Therefore, $K(x) > \frac{n}{8} - O(\log n)$. But this contradicts the well-known fact [13] that there is no algorithm that for any given n finds a string of complexity at least $f(n)$ where f is a computable total unbounded function.

(ii) The proof is by contradiction. Assume Item (ii) is false. Fix a large length $n = 2^k$ and let $A_1 = \{0,1\}^n$, so that $\alpha = 2 \log k \geq K(A_1)$. Let x be a string of length n, let $N^\alpha < 2^{\alpha+1}$ be the number of halting programs of length at most α, and let $\mathcal{A} = \{A_1, \ldots, A_m\}$ be the set of all finite sets of complexity at most α. Since $x \in A_1$, the value $\beta_x(\alpha)$ is finite and $\beta_x(\alpha) = \min_{A \in \mathcal{A}} \{\delta(x \mid A)\}$. Assuming β_x is upper semi-computable, we can run the following algorithm:

Algorithm: Given N^α, α, and x,

Step 1: Enumerate all finite sets $\mathcal{A} = \{A_1, \ldots, A_m\}$ of complexity $K(A_i) \leq \alpha$. Since we are given N^α, α we can list them exhaustively.

Step 2: Dovetail the following computations simultaneously:

Step 2.1: Upper semi-compute $\beta_x(\alpha)$, for all x of length n.

Step 2.2: For all $i = 1, \ldots, m$, lower semi-compute $\delta(x \mid A_i) = \log |A_i| - K(x \mid A_i)$.

We write the approximations at the tth step as $\beta_x^t(\alpha)$, $\delta^t(x \mid A_i)$, and $K^t(x \mid A_i)$, respectively. We continue the computation until step t such that

$$\beta_x^t(\alpha) < \min_{A \in \mathcal{A}} \{\delta^t(x \mid A)\} + n/\log^4 n.$$

This t exists by the assumption above. By definition, $\min_{A \in \mathcal{A}} \{\delta(x \mid A)\} = \beta_x(\alpha) \leq \beta_x^t(\alpha)$. Let A^x denote the set minimizing the right-hand side. (Here we use that x belongs to a set in $\mathcal{A}$.) Together, this shows that $\log |A^x| - \beta_x^t(\alpha) \leq K(x \mid A^x)$ and $\log |A^x| - \beta_x^t(\alpha) \geq K^t(x \mid A^x) - n/\log^4 n \geq K(x \mid A^x) - n/\log^4 n)$. Thus we obtained an estimation $\log |A^x| - \beta_x^t(\alpha)$ of $K(x \mid A^x)$ with precision $n/\log^4 n$. We use that $K(x \mid A^x)$ is a good approximation to $K(x)$:

$$\begin{aligned} K(x \mid A^x) - c_1 \leq K(x) &\leq K(x \mid A^x) + |A^x| + c_1 \\ &\leq K(x \mid A^x) + \alpha + c_1, \end{aligned}$$

where c_1 is a constant. Consequently,

$$K(K(x) \mid x) \leq K(N^\alpha, \alpha, K(x) - \log |A^x| + \beta_x^t(\alpha)) + c_2.$$

where the constant c_2 is the length of a program to reconstruct α, N^α and $K(x) - \log |A^x| + \beta_x^t(\alpha) \leq \alpha + c_1 + n/\log^4 n$, and combining this information with the conditional information x, to compute $K(x)$. Observing $K(N^\alpha) = \alpha - K(\alpha) + O(1)$ by [10], and substituting $\alpha = 2 \log \log n$, there is a constant c_3 such that

$$K(K(x) \mid x) \leq 2 \log \log n + \log n - 4 \log \log n + c_3.$$

However, for every n, we can choose an x of length n such that $K(K(x) \mid x) \geq \log n - \log \log n$ by [8], which gives the required contradiction. ■

Open question. Is there a non-increasing (with respect to α) upper semi-computable function $f_x(\alpha)$ such that, for all x, $\beta_x(\alpha) = \mathcal{E}(f_x(\alpha))$ for $\varepsilon = \delta = O(\log |x|)$ (or for $\varepsilon = \delta = o(|x|)$)?

Acknowledgment

We thank Harry Buhrman, Marcus Hütter, Andrei A. Muchnik, Osamu Watanabe, for helpful discussions, and Tom Cover, Péter Gács, and Leonid A. Levin for historical

details about Kolmogorov's original proposal of the structure function in the early seventies and about their own unpublished work on the subject. After the first author questioned Muchnik whether the result in Theorem IV.8 was perhaps known before, the latter answered negatively and supplied an independent proof of it. He also suggested and proved Theorem B.5. Watanabe suggested the question treated in Remark IV.12.

References

[1] A.R. Barron and T.M. Cover, Minimum Complexity Density Estimation, *IEEE Trans. Inform. Theory*, 37(1991), 1034–1054.
[2] A.R. Barron, J. Rissanen, and B. Yu, The minimum description length principle in coding and modeling, *IEEE Trans. Inform. Theory*, IT-44:6(1998), 2743–2760.
[3] T.M. Cover, Kolmogorov complexity, data compression, and inference, pp. 23–33 in: *The Impact of Processing Techniques on Communications*, J.K. Skwirzynski, Ed., Martinus Nijhoff Publishers, 1985.
[4] T.M. Cover, Attending [16], Email to PV, January 24, 2002.
[5] T.M. Cover, P. Gács and R.M. Gray, Kolmogorov's contributions to Information Theory and Algorithmic Complexity, *Ann. Probab.*, 17(1989), 840–865.
[6] T.M. Cover and J.A. Thomas, *Elements of Information Theory*, Wiley, New York, 1991.
[7] R. A. Fisher, On the mathematical foundations of theoretical statistics, *Philosophical Transactions of the Royal Society of London, Ser. A*, 222(1922), 309–368.
[8] P. Gács, On the symmetry of algorithmic information, *Soviet Math. Dokl.*, 15 (1974) 1477–1480. Correction: ibid., 15 (1974) 1480.
[9] P. Gács, Attending [16], Email to PV, January 24, 2002.
[10] P. Gács, J. Tromp, P.M.B. Vitányi. Algorithmic statistics, *IEEE Trans. Inform. Th.*, 47:6(2001), 2443–2463.
[11] Q. Gao, M. Li and P.M.B. Vitányi, Applying MDL to learn best model granularity, *Artificial Intelligence*, 121(2000), 1–29.
[12] L.A. Levin. Emails to PV on February 7,11,20, 2002.
[13] M. Li and P.M.B. Vitányi, *An Introduction to Kolmogorov Complexity and its Applications* , Springer-Verlag, New York, 2nd Edition, 1997.
[14] A.N. Kolmogorov, Three approaches to the quantitative definition of information, *Problems Inform. Transmission* 1:1 (1965) 1–7.
[15] A.N. Kolmogorov. Complexity of Algorithms and Objective Definition of Randomness. A talk at Moscow Math. Soc. meeting 4/16/1974. An abstract available in *Uspekhi Mat. Nauk* 29:4(1974),155; translation in Section I.
[16] A.N. Kolmogorov, Talk at the Information Theory Symposium in Tallinn, Estonia, 1974, according to [9], [4].
[17] P. Martin-Löf, The definition of random sequences, *Inform. Contr.*, 9(1966), 602-619.
[18] J.J. Rissanen, A Universal Prior for Integers and Estimation by Minimum Description Length, *Annals of Statistics*, 11:2(1983), 416–431.
[19] J.J. Rissanen, Kolmogorov's Structure Function for Probability Models, Proc. IEEE Information Theory Workshop, IEEE Press, 2002, 98–99.
[20] C.E. Shannon, The mathematical theory of communication, *Bell System Tech. J.*, 27(1948), 379–423, 623–656.
[21] A.Kh. Shen, The concept of (α, β)-stochasticity in the Kolmogorov sense, and its properties, *Soviet Math. Dokl.*, 28:1(1983), 295–299.
[22] A.Kh. Shen, Discussion on Kolmogorov complexity and statistical analysis, *The Computer Journal*, 42:4(1999), 340–342.
[23] P.M.B. Vitányi and M. Li, Minimum Description Length Induction, Bayesianism, and Kolmogorov Complexity, *IEEE Trans. Inform. Theory*, IT-46:2(2000), 446–464.
[24] P.M.B. Vitányi, Meaningful information, Los Alamos National Laboratories Archives, http://xxx.lanl.gov/abs/cs.CC/0111053
[25] V.G. Vovk, Learning about the parameter of the Bernoulli Model, *J. Comput. System Sci.*, 55(1997), 96–104.
[26] 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.
[27] V.V. V'yugin, Algorithmic complexity and stochastic properties of finite binary sequences, *The Computer Journal*, 42:4(1999), 294–317.
[28] V.V. V'yugin. Does snooping help? *Theoret. Comput. Sci.*, 276:1/2(2002), 407–415.
[29] C.S. Wallace and P.R. Freeman, Estimation and inference by compact coding, *J. Royal Stat. Soc., Series B*, 49 (1987) 240-251; Discussion: ibid.,252-265.