Identification of Probabilities of Languages [1]

Paul M.B. Vitányi and Nick Chater

Abstract

We consider the problem of inferring the probability distribution associated with a language, given data consisting of an infinite sequence of elements of the languge. We do this under two assumptions on the algorithms concerned: (i) like a real-life algorothm it has round-off errors, and (ii) it has no round-off errors. Assuming (i) we (a) consider a probability mass function of the elements of the language if the data are drawn independent identically distributed (i.i.d.), provided the probability mass function is computable and has a finite expectation. We give an effective procedure to almost surely identify in the limit the target probability mass function using the Strong Law of Large Numbers. Second (b) we treat the case of possibly incomputable probabilistic mass functions in the above setting. In this case we can only pointswize converge to the target probability mass function almost surely. Third (c) we consider the case where the data are dependent assuming they are typical for at least one computable measure and the language is finite. There is an effective procedure to identify by infinite recurrence a nonempty subset of the computable measures according to which the data is typical. Here we use the theory of Kolmogorov complexity. Assuming (ii) we obtain the weaker result for (a) that the target distribution is identified by infinite recurrence almost surely; (b) stays the same as under assumption (i). We consider the associated predictions.

I. Introduction

In cognition and science one learns by observation. The perceptual system of an individual person, or the data-gathering resources of a scientific community, incrementally gathers empirical data, and attempts to find the structure in that data. The question arises: under what conditions is it possible *precisely* to infer the structure underlying those observations? Or, relatedly, under what conditions could a machine learning algorithm potentially precisely recover this structure? We can model this problem as having

Vitányi is with the national research institute for mathematics and computer science in the Netherlands (CWI) and the University of Amsterdam. Address: CWI, Science Park 123, 1098 XG, Amsterdam, The Netherlands. Email: paulv@cwi.nl.

Chater is with the Behavioural Science Group. Address: Warwick Business School, University of Warwick, Coventry, CV4 7AL, UK. Email: Nick.Chater@wbs.ac.uk. Chater was supported by ERC Advanced Grant "Cognitive and Social Foundations of Rationality."

the following form: given a semi-infinite sequence of samples from a probability distribution, under what conditions is it possible precisely to recover this distribution? Moreover, we can focus on the case where each observation is coded in a language—that is, each observation corresponds to an element of a countable set of sentences. Then the problem at hand is to recover the probability induced by the language.

In the context of the cognitive processes of an individual, information from the senses is presumed to be coded in the brain to some finite precision (indeed, neural firing is discrete, [19]). Linguistic input, in particular, can be coded in a hierarchy of discrete symbolic representations, as described by generative grammar (for example, [12]). And, in the context of the operation of the scientific community, data is digitally coded to finite precision in symbolic codes. There are at least three reasons for scepticism that precisely recovering the probability of possible observations is possible.

First, as observed by Popper [17], however much data has been encountered, any theory or model can be falsified by the very next piece of data. However many white swans are observed, there is always the possibility that the very next swan will be black, or some more unlikely color. If, in the case of a child learning a language, however often the child encounters sentences following a particular set of grammatical rules, it is always possible that the very next sentence encountered will violate these rules (for example [16]). Thus, however, much data been encountered, there is no point at which the learner can announce a particular probability as correct with any certainty. But this does not rule out the possibility that the learner might learn to identify the correct probability in the limit. That is, perhaps the learner might make a sequence of guesses, finally locking on to correct probability and sticking to it forever—even though the learner can never know for sure that it has identified the correct probability successfully. We shall therefore consider identification in the limit below (following, for example, [5], [8], [16]).

Second, in conventional statistics, probabilistic models are typically idealized as having continuous valued parameters; and hence there is an uncountable number of possible probabilities, from which the correct probability is to be recovered. In general it is impossible that a learner can make a sequence of guesses that precisely locks on to the correct values of continuous parameters. This, since the possible strategies of learners are effective in the sense of Turing [20] and thus countable. This assumption is, of course, obeyed by any practical machine learning algorithm and we assume also by the brain. The set of such strategies can express only a countable number of possible hypotheses. From this mild assumption, it is, of course, immediately evident that the overwhelming majority of a continuum of hypotheses cannot be represented, let alone learned. Moreover, there is a particularly natural restriction

concerning the set of probabilites from which the learner chooses: it must be *computable* (these notions are made precise below). This seems a reasonable restriction, both in cognitive science and scientific methodology. After all, the assumption that individual human information processing is computable is a founding assumption of cognitive science (see for example, [18]); and the same constraint arguably applies to every practically usable scientific theory (although whether this holds is discussed in [2]). We shall see below that restricting ourselves to the computable simplifies the problem of precisely reconstructing the correct probability from the observed data. For example, the computability of the set of possible probabilities means that these can be enumerated; and it may then be possible to gradually home in the correct probability, by successively eliminating earlier ones in the enumerated list. As we see below, it is also possible to provide approximation results if the computability restriction is dropped.

A third reason for initial scepticism also concerns computability—this time for the *learner*, not just the probability to be recovered. Even if there is, in principle, sufficient data to pin down the correct probability precisely, there remains the question of whether there is a feasible computational procedure that can reliably map from data to a sequence of guesses that eventually lock on to the correct probability. Real-life computational procedures are finite and always have round-off errors. We outline positive results that can be obtained for computable learners with or without such round-off errors.

A. Preliminaries

A language is a set of sentences. The learnability of a language under various computational assumptions is the subject of an immensely influential approach in [4] and especially [5]. But surely in the real world the chance of one sentence of a language being used is different from another one. For example, many short sentences have a larger chance of turning up than very long sentences. Thus, the elements of a given language are distributed in a certain way. There arises the problem of identifying or approximating this distribution.

We first introduce some terminology. A function is *computable*, if there is a Turing machine (or any other equivalent computational device such as a universal programming language) that maps the arguments to the values. We say that we *identify* a function f *in the limit* if we effectively produce an infinite sequence $f_1, f_2, \ldots$ of functions and $f_i = f$ for all but finitely many i. This corresponds to the notion of "identification in the limit" in [5]. We *identify* a function f *by infinite recurrence* if $f_i = f$ for infinitely many i. A sequence of functions *converges to* a function f *pointswize* if $\lim_{i\to\infty} f_i(a) = f(a)$ for all a in the domain of f. The functions we are interested in are versions of the probability mass

functions.

The restriction to computable probability mass functions (or more generally any restriction to a countable set of probability mass functions) is both cognitively realistic (if we assume language is generated by a computable process) and dramatically simplifies the problem of language identification. This is also the case for the use of algorithms with round-off errors.

B. Related work

In [1] (citing previous more restricted work) a target probability mass function was identified in the limit when the data are drawn independent identically distributed (i.i.d.) in the following setting. Let the target probability mass function p be an element of a list $q_1, q_2, \ldots$ subject to the following conditions: (i) every $q_i : \mathcal{N} \to \mathcal{R}^+$ is a probability mass function where $\mathcal{N}$ and $\mathcal{R}^+$ denote the positive natural numbers and the positive real numbers, respectively; (ii) there is a total computable function $C(i, x, \epsilon) = r$ such that $(q_i(x) - r) \leq \epsilon$ with $r, \epsilon > 0$ are rational numbers. The technical means used are the Law of the Iterated Logarithm and the Kolmogorov-Smirnov test. The algorithms used have no round-off errors. However, the list $q_1, q_2, \ldots$ cannot contain all computable probability mass functions, Lemma 4.3.1 in [14].

C. Results

In Section II we deal with probability mass functions. For technical reasons we introduce a weaker form thereof called "semiprobability mass functions." Consider a probability mass function satisfying (A.1) below associated with a language. The data consist of an infinite sequence of elements of this language that are drawn i.i.d. The aim is to identify the probability mass function given the data. (In contrast to [1] we allow all computable probability mass functions.) In Section II-A we consider algorithms without round-off errors. Then, we identify the target distribution by infinite recurrence almost surely. In Section II-B the identification algorithm is subject to round-off errors and we identify the target distribution in the limit almost surely (underpinning the result announced in [7]). In Section III we treat the case of possibly uncomputable probability mass functions. Then we can only show pointswize convergence almost surely. This result holds both with or without round-off errors. The technical tool in these sections is the Strong Law of the Large Numbers. In all these results the language concerned can be infinite. In Section IV we consider the case where the data are dependent assuming they are typical (Definition 9) for at least one computable measure. In contrast to the i.i.d. case, it is possible that the data are typical for many

measures. The language concerned is finite and the identification algorithm has round-off errors. Then, we identify by infinite recurrence (possibly a subset) of computable measures for which the data are typical. The technical tool is Kolmogorov complexity. Finally, In Section V we consider the associated predictions. We defer the proofs of the theorems to Appendix A.

II. Computable Probability Mass Functions

Most known probability mass functions are computable provided their parameters are computable. In order that it is computable we only require that the probability mass function is finitely describable and there is an effective process producing it [20].

It is known that the overwhelming majority of real numbers are not computable. An example of an incomputable probability mass function therefore is the one associated with a biased coin with an incomputable probability p of outcome heads and probability $1-p$ of outcome tails, $0 \leq p \leq 1$. On the other hand, if p is lower semicomputable, then we can effectively find nonnegative integers $a_1, a_2, \ldots$ and $b_1, b_2, \ldots$ such that $a_n/b_n \leq a_{n+1}/b_{n+1}$ and $\lim_{n\to\infty} a_n/b_n = p$. Let us generalize this observation.

Definition 1: If a function has as values pairs of nonnegative integers, such as (a, b), then we can interpret this value as the rational a/b. This leads to the notion of a computable function with rational arguments and real values. A real function $f(x)$ with x rational is *semicomputable from below* if it is defined by a rational-valued total and computable function $\phi(x, k)$ with x a rational number and k a nonnegative integer such that $\phi(x, k+1) \geq \phi(x, k)$ for every k and $\lim_{k\to\infty} \phi(x, k) = f(x)$. This means that f can be computably approximated arbitrary close from below (see [14], p. 35). A function f is *semicomputable from above* if $-f$ is semicomputable from below. If a real function is both semicomputable from below and semicomputable from above then it is *computable*. A function f is a *semiprobability* mass function if $\sum_x f(x) \leq 1$ and it is a *probability* mass function if $\sum_x f(x) = 1$. It is customary to write $p(x)$ for $f(x)$ if the function involved is a semiprobability mass function.

We cannot effectively enumerate all computable probability mass functions (this is a consequence of Lemma 4.3.1 in [14]). However, it is possible to enumerate all and only the semiprobability mass functions that are lower semicomputable. This is done by fixing an effective enumeration of all Turing machines of the so-called *prefix* type. (Such an enumeration is quite the same as effectively enumerating all programs in a conventional computer programming language that is computationally universal and were the programs are prefix-free—a set is prefix-free if no element is a proper prefix of any other. Most,

if not all, conventional computer programming languages satisfy these requirements.) It is possible to change every Turing machine description in the enumeration into one that computes a semiprobability mass function that is computable from below, Theorem 4.3.1 in [14] (originally in [21], [13]). The result is

$$\mathcal{Q} = q_1, q_2, \ldots, \tag{II.1}$$

a list containing all and only semiprobability mass functions that are semicomputable from below. Without loss of generality every element of $\mathcal{Q}$ is over the alphabet L.

DEFINITION 2: There is a total and computable function $\phi(i, x, t) = q_i^t(x)$ such that $\phi(i, x, t) \leq \phi(i, x, t+1)$ and $\lim_{t\to\infty} q_i^t(x) = q_i(x)$.

Every probability mass function is a semiprobability mass function, and every computable probability mass function is semicomputable from below. Therefore, every computable probability mass function is in list $\mathcal{Q}$. Indeed, every such function will be in the list infinitely often, which follows simply from the fact that there are infinitely many computer programs that compute a given function. If a lower semicomputable semiprobability mass function is a probability mass function, then it must be computable, [14] Example 4.3.2. Therefore, every probability mass function in the list is computable. It is important to realize that, although the description of every computable probability mass function is in list $\mathcal{Q}$, it may be there in lower semicomputable format and we may not know it is computable.

A. Algorithms Without Round-Off Errors

DEFINITION 3: Let $x = x_1, x_2, \ldots$ be an infinite sequence of elements of the language L i.i.d. drawn according to a computable probability mass function p. Let $Q(p)$ be defined as the set of indices of elements of $\mathcal{Q}$ that are copies of p.

THEOREM 1: COMPUTABLE I.I.D. PROBABILITY IDENTIFICATION (NO ROUND-OFF) Let L be a language $\{a_1, a_2, \ldots\}$ (a countably finite or infinite set) with a computable probability mass function p. Without loss of generality we assume that every $a \in L$ is a finite integer. Let the mean of p exist ($\sum_{a\in L} ap(a) < \infty$). The algorithm in the proof takes as input an infinite sequence $x = x_1, x_2, \ldots$ of elements of L drawn i.i.d. according to p. After processing x_n the algorithm computes as output the index i_n of a semiprobability mass function in the enumeration $\mathcal{Q}$. Define Q_∞ as the set of indices of elements in $\mathcal{Q}$ that appear infinitely often in the sequence produced by the algorithm. Then,

(i) $Q_\infty \neq \varnothing$ almost surely;

(ii) if $i \in Q_\infty$ then $i \in Q(p)$ almost surely; and

(iii) almost surely $\liminf_{n\to\infty} i_n = \min Q(p)$.

B. Algorithms With Round-Off Errors

We will want to computationally separate probability mass functions p for which $\sum_x p(x) = 1$ from semiprobability mass functions q for which $\sum_x q(x) < 1$ in the list $\mathcal{Q}$.

DEFINITION 4: To deal with truncation errors in the above (in)equalities we use the following notation. The truncation error is a known additive term $\pm\epsilon$, $\epsilon > 0$, and we denote (in)equalities up to this truncation error by $\overset{+}{\leq}$, $\overset{+}{\geq}$, $\overset{\pm}{=}$, $\overset{+}{<}$, and $\overset{+}{>}$. Every function that satisfies the probability mass function equality within the round-off error is viewed as a probability mass function.

Let L be a language $\{a_1, a_2, \ldots\}$, $\#a(x_1, x_2, \ldots, x_n)$ be the number of elements in $x_1, x_2, \ldots, x_n$ equal $a \in L$, and k' be the least index of an element q_i in the list $\mathcal{Q}$ such that

$$\lim_{n\to\infty} \sum_{j=1}^{n} \left| q_i(a_j)^n - \frac{\#a_j(x_1, x_2, \ldots, x_n)}{n} \right| \overset{\pm}{=} 0.$$

THEOREM 2: COMPUTABLE I.I.D. PROBABILITY IDENTIFICATION (ROUND-OFF) Let L be a language $\{a_1, a_2, \ldots\}$ (a countably finite or infinite set) with a computable probability mass function p. Without loss of generality we assume that every $a \in L$ is a finite integer. Let the mean of p exist ($\sum_{a\in L} ap(a) < \infty$). The algorithm in the proof takes as input an infinite sequence $x = x_1, x_2, \ldots$ of elements of L drawn i.i.d. according to p. After processing x_n the algorithm computes as output the index i_n of a semiprobability mass function in the enumeration $\mathcal{Q}$. There exists an N such that $i_n = k'$ for all $n \geq N$. We have $k' \leq k$ with k as in Theorem 1.

III. GENERAL PROBABILITY MASS FUNCTIONS

Can we get rid of the restriction that the probability mass function be computable? Above we used the computability to consider a well-ordered countable list of lower semicomputable semiprobability mass functions, and pin-pointed the least occurrence of the target in the list. We use the fact that we have a guarantee that the target is in the list. If we have no such guarantee (possibly there is no list), we can still converge pointswise to the *empirical* probability mass function of the language L based on the data $x_1, x_2, \ldots$. We do this by an algorithm computing the probability $p(a)$ in the limit for all $a \in L$.

Note that this is quite different from Theorems 1, 2. There we indicated (the least occurrence of) p precisely in a well-ordered list even though the result only holds "almost surely." In contrast, here we find

$p(a)$ for all $a \in L$ in the limit only. Moreover, this holds also "almost surely." However, the probability mass functions considered here consist of *all* probability mass functions on L—computable or not.

THEOREM 3: I.I.D. PROBABILITY APPROXIMATION WITH AN ALGORITHM WITHOUT ROUND-OFF ERROR Let L be a language $\{a_1, a_2, \ldots\}$ (a countably finite or infinite set) with a computable probability mass function p. Without loss of generality we assume that every $a \in L$ is a finite integer. Let the mean of p exist ($\sum_{a \in L} ap(a) < \infty$). The algorithm in the proof takes as input an infinite sequence $x_1, x_2, \ldots$ of elements of L drawn i.i.d. according to p. After processing x_n the algorithm computes as output a probability mass function p_n. Almost surely for all $a \in L$ the $\lim_{n\to\infty} p_n(a) = p(a)$.
With round-off error this theorem is about the same.

REMARK 1: Can this result be strengthened to a form of dependent variables? It all depends on whether the Strong Law of Large Numbers holds. We know from [3], p. 474, that the Strong Law of Large Numbers holds for stationary ergodic sources with finite expected value. $\diamond$

IV. COMPUTABLE MEASURES

This time let the language $L = \{a_1, a_2, \ldots, a_m\}$ be a finite set, and $x_1, x_2, \ldots$ the data consisting of an infinite sequence of elements from L. We drop the requirement of independency of the different elements in the data. Thus, we assume that our data sequence $x_1, x_2, \ldots$ is possibly dependent. This implies that the probability model for L is more general than i.i.d.. In fact, we will allow all computable measures on the infinite sequences of elements from L. Thus, the probability model used includes stationary processes, ergodic processes, Markov processes of any order, and other models, provided they are computable.

Given a finite sequence $x = x_1, x_2, \ldots, x_n$ of elements of the basic set L, we consider the set of infinite sequences starting with x. The set of all such sequences is written as Γ_x, the *cylinder* of x. We associate a probability $\mu(\Gamma_x)$ with the event that an element of Γ_x occurs. Here we abbrieviate $\mu(\Gamma_x)$ to $\mu(x)$. The transitive closure of the intersection, union, complement, and countable union of cylinders gives a set of subsets of L^∞. The probabilities associated with these subsets are derived from the probabilities of the cylinders in standard ways [9]. A *semimeasure* μ satisfies the following:

$$\mu(\epsilon) \leq 1 \tag{IV.1}$$
$$\mu(x) \geq \sum_{a \in L} xa,$$

and if equality holds instead of each inequality we call μ a *measure*. Using the above notation, a semimeasure μ is *lower semicomputable* if it is defined by a rational-valued computable function

$\phi(x,k)$ with $x \in L^*$ and k a nonnegative integer such that $\phi(x,k+1) \geq \phi(x,k)$ for every k and $\lim_{k\to\infty} \phi(x,k) = \mu(x)$. This means that μ can be computably approximated arbitrary close from below for each argument $x \in L^*$.

To separate measures from semimeasures that are not measures using an algorithm subject to round-off errors, we have to deal with truncation errors in the (semi)measure (in)equalities. This truncation error is a known additive term $\pm\epsilon$, $\epsilon > 0$. Again we use Definition 4: (in)equalities up to this truncation error are denoted by $\overset{+}{\leq}$, $\overset{+}{\geq}$, $\overset{+}{=}$, $\overset{+}{<}$, and $\overset{+}{>}$.

In the argument below we want to effectively enumerate all computable measures. By Lemma 4.5.1 of [14], if a lower semicomputable semimeasure is a measure, then it is computable. Thus it suffices to effectively enumerate all lower semicomputable measures. By Lemma 4.5.2 of the cited reference this is not possible. But if we effectively enumerate all lower semicomputable semimeasures, then this enumeration includes all lower semicomputable measures which are a fortiori computable measures. This turns out to be possible (originally in [21], [13]). Just as in the case of lower semicomputable semiprobabilities, but with a little more effort, we can effectively enumerate all and only lower semicomputable semimeasures, as is described in the proof of Theorem 4.5.1 of [14]. This goes by taking a standard enumeration of all Turing machines T_1, T_2, ... of the so-called *monotone* type. Subsequently we transform every Turing machine in the list to one that lower semicomputes a semimeasure. Also, it is shown that all lower semicomputable semimeasures are in the list. The result is

$$\mathcal{M} = \mu_1, \mu_2, \ldots. \tag{IV.2}$$

This list contains all and only semimeasures that are semicomputable from below. It is important to realize that, although the description of every computable measure is in list $\mathcal{M}$, it may be there in lower semicomputable format and we may not know it is computable.

DEFINITION 5: There is a total and computable function $\phi(i,x,t) = \mu_i^t(x)$ such that $\phi(i,x,t) \leq \phi(i,x,t+1)$ and $\lim_{t\to\infty} \mu_i^t(x) = \mu_i(x)$.

REMARK 2: Let our data be $x_1, x_2, \ldots$. Possibly there are none or more than one computable measures that have these data as a "typical" sequence. Here we use "typicality" according to the Definition 9 below. Such typical infinite sequences are also called "random" with respect to the measure involved. For instance, let the data sequence be $0, 0, \ldots$. Then this is a typical sequence of the computable measure μ_{i_1} that gives measure 1 to every initial segment of this data sequence. But if we consider μ_{i_2} that gives measure $\frac{1}{2}$ to every initial segment of $0, 0, \ldots$ and also measure $\frac{1}{2}$ to every initial segment of $1, 1, \ldots$,

then the considered data sequence is also a typical sequence of μ_{i_2}. Similarly, it is a typical sequence of the measure μ_{i_k} that gives measure $1/k$ to every initial segment of $0, 0, \ldots$ through $k-1, k-1, \ldots$ ($k < \infty$).

Hence our task is not to identify the measure according to which the data sequence was generated as a typical sequnce, but to identify measures which *could* have generated the data sequence as a typical sequence. Note that we have reduced our task from identifying *the* computable measure generating the data, which is not possible, to *some* computable measures that *could* have generated the data as a typical sequence. $\diamond$

REMARK 3: We assume here that L is finite. This is no genuine restriction since all real natural or artificial languages that ever existed contain less than, say, 10^{100} elements. This is supported by the fact that 10^{100} far exceeds the number of atoms currently believed to exist in the observable universe. We assume that L is finite since it makes the computation of $\sum_{a \in L} xa$ for $x \in L^*$ effective.

Where appropriate, we shall use the (in)equalities according to Definition 4. This has bad and good effects. The bad effect is that semimeasures that violate the measure equalities by at most the truncation error are counted as measures. The good effects are, besides computationally verifiable (in)equalities, that in the infinite processes to construct lower semicomputable semimeasures (the proof of Theorem 4.5.1 of [14]) we only need to consider finite initial segments. $\diamond$

Let L be a language with a computable measure μ on the infinite sequences of its elements. We recall the notion that an infinite sequence x is "typical" for μ.

DEFINITION 6: Let $x = x_1, x_2, \ldots$ be an infinite sequence of elements of the language L. The infinite sequence x is *typical* or *random* for a computable measure μ if it passes all effective sequential tests for randomness with respect to μ in the sense of Martin-Löf [15]. The set of such sequences have μ-measure one. We define $M(x)$ as the set of indices of elements of the list $\mathcal{M}$ that are computable measures μ such that x is typical for μ.

THEOREM 4: COMPUTABLE MEASURE IDENTIFICATION BY ALGORITHMS WITH ROUND-OFF ERRORS Let $L = \{a_1, a_2, \ldots, a_m\}$, $m < \infty$, be a language and let $x_1, x_2, \ldots$ with $x_i \in L$ be an infinite data sequence. Assume that the data sequence is typical for at least one computable measure. The algorithm in the proof takes as input the infinite sequence $x_1, x_2, \ldots$. After processing x_n the algorithm computes as output the index i_n of an element of $\mathcal{M}$. Define M_∞ as the set of indices of elements in $\mathcal{M}$ that appear infinitely often in the sequence produced by the algorithm. Then, $M_\infty \subseteq M(x)$, $M_\infty \neq \varnothing$, and $|M_\infty| < \infty$.

V. PREDICTION

In Sections II and III the data are drawn i.i.d. according to a probability mass function p on the elements of L. Given p, we can predict the probability $p(a|x_1, \ldots, x_n)$ that the next draw results in an element a when the previous draws resulted in $x_1, \ldots, x_n$. (The same holds in appropriate form for a good pointswise approximation $\hat{p}$ of p.)

For measures as in Section IV, allowing dependent data, the situation is quite different. In the first place there can be many measures that have $x = x_1, x_2, \ldots$ as typical (random) data. In the second place, different of these measures may give different probability predictions using the same initial segment of x.

Let us give a simple example. Suppose the data is $x = a, a, \ldots$. This data is typical for the measure μ_1 defined by $\mu_1(x) = 1$ for every x consisting of a finite or infinite string of a's and $\mu_1(x) = 0$ otherwise. But the data is also typical for μ_2 which gives probability $\mu_2(x) = \frac{1}{2}$ for every string consisting of an a followed by a finite or infinite string of a's, or a followed by a finite or infinite string of b's.

Firstly, μ_1 is not equal to μ_2, even though x is typical for both of them. Secondly, $\mu_1(a|a) = 1$. But $\mu_2(a|a) = \mu_2(b|a) = \frac{1}{2}$. In fact, $\mu_1(y|a) = 1$ for every y consisting of a finite or infinite string of a's, and 0 otherwise. The conditional probability $\mu_2(y|a)$ is $\frac{1}{2}$ for y consisting of a finite or infinite string of a's or y consisting of a finite or infinite string of b's, and 0 otherwise. Thus, different measures for which the data is typical may give very different predictions. With respect to predictions we can only proceed as follows: (i) find one or more measures for which the data is typical, and (ii) predict according to one of these measures that we select.

It does not seem make sense to make a weighted prediction according to the measures for which the data is typical. There may not be a single measure among them making that prediction. Moreover, the consecutive data resulting from many predictions may not be typical with respect to any of the original measures.

The question arises how the i.i.d. case and the measure case relate to one another. The answer is as follows. For ease of writing we will ignore the adjective "computable." It is clear that the i.i.d. probabilities are a subset of the more general case of measures. Containment is proper since there is a measure that is not an infinite sequence of i.i.d. draws according to a probability mass function. An example is given below.

Moreover, an infinite sequence of data can be typical for more than one measure, even though if such a sequence is typical for an infinite sequence of i.i.d. draws of any probability mass function, then it

is typical for an infinite sequence of i.i.d. draws of only this single probability mass function. Thus, if the data is typical for different measures, then only one of the measures involved is a probability mass function.

Let us give an example. The measure μ_1 above has a single typical sequence $x = a, a, \ldots$. This measure results from infinitely many i.i.d. draws of L according to the probability mass function $p(a) = 1$ and $p(z) = 0$ for $z \in L \setminus \{a\}$. But x is also typical for μ_2, a measure which has no i.i.d. case that corresponds to it. This can be seen as follows. The measure μ_2 has also the typical sequence $y = a, b, b, \ldots$, a sequence such that no infinite number of i.i.d. draws of any probability mass function corresponds to it. Namely, the probability of a and b must both be non-zero to yield the sequence y. Hence a typical infinite sequence must contain (in the i.i.d. case) infinitely a's and b's. But y does not do so.

Thus, the i.i.d. case is a proper subset of the measure case. A single infinite sequence can be typical for many (infinitely many) measures, even though if it is typical for an i.i.d. case it is typical for only a single one of the probability mass functions. But there are infinite sequences that are typical for many measures but not typical for any case of infinitely many i.i.d. draws according to a probability mass function.

APPENDIX

Proof: OF THEOREM 1. Our data is, by assumption, generated by i.i.d. draws according to a computable probability mass function p satisfying (A.1). Formally, the data $x_1, x_2, \ldots$ is generated by a sequence of random variables $X_1, X_2, \ldots$, each of which is a copy of a single random variable X with probability mass function $P(X = a) = p(a)$ for every $a \in L$. Without loss of generality $p(a) > 0$ for all $a \in L$. The mean of X exists by (A.1).

REMARK 4: In probability theory the statement *almost surely* means "with probability one." Let us illustrate this notion. It is possible that a fair coin generates an infinite sequence $0, 0, \ldots$ even though the probability of 1 at each trial is $\frac{1}{2}$. The uniform measure of the set of infinite sequences, such that the relative frequency of 1's goes to the limit $\frac{1}{2}$, is one. Call sequences in that set *pseudo typical*. The probability that an infinite sequence is pseudo typical is one, even though there are infinite sequences (like in the example above) that are not pseudo typical. Thus, "almost surely" may not mean "with certainty." $\diamond$

The Strong Law of Large Numbers (originally in [10]) states that if we perform the same experiment a large number of times, then almost surely the average of the results goes to the expected value. That

is, if a mild condition is satisfied. We require that our sequence of random variables $X_1, X_2, \ldots$ satisfies Kolmogorov's criterion that

$$\sum_i \frac{(\sigma_i)^2}{i^2},$$

where σ_i^2 is the variance of X_i in the sequence of mutually independent random variables $X_1, X_2, \ldots$. Since all X_i's are copies of a single X, all X_i's have a common distribution p. In this case we use the theorem on top of page 260 in [6]. To apply the Strong Law in this case it suffices that the mean of X exists. (We denote this mean by μ, not to be confused with the notation of measures $\mu(\cdot)$ we use below.) That is, we require that

$$\mu = \sum_{a \in L} a p(a) < \infty. \tag{A.1}$$

Then, the Strong Law states that

$$\Pr\left(\lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} X_i = \mu \right) = 1,$$

or $(1/n)\sum_{i=1}^{n} X_i$ converges almost surely to μ as $n \to \infty$.

To determine the probability of an $a \in L$ we consider the related random variables X_a with just two outcomes $\{a, \bar{a}\}$. This X_a is a Bernoulli process $(q, 1-q)$ where $q = p(a)$ is the probability of a and $1 - q = \sum_{b \in L \setminus \{a\}} p(b)$ is the probability of $\bar{a}$. If we set $\bar{a} = \min(L \setminus \{a\})$ while the probability of $\bar{a}$ is $1 - q = \sum_{b \in L \setminus \{a\}} p(b)$, then the mean μ_a of X_a is

$$\mu_a = aq + \bar{a}(1-q) \leq \mu.$$

REMARK 5: Recall that L may be infinite, that is, $L = \{a_1, a_2, \ldots, \}$. Then a priori it could be that $\lim_{j\to\infty} \mu_{a_j} = \infty$. But we have just proved that not only this does not happen but even $\mu_{a_j} \leq \mu$ for every j. $\diamond$

Thus, every $a \in L$ incurs a random variable X_a for which the equivalent of (A.1) applies. Therefore, according to the cited theorem the quantity $(1/n)\sum_{i=1}^{n} (X_a)_i$ converges almost surely to μ_a as $n \to \infty$. Therefore, almost surely

$$\lim_{n\to\infty} \sum_{j=1}^{n} \left| p(a_j) - \frac{\#a_j(x_1, x_2, \ldots, x_n)}{n} \right| = 0. \tag{A.2}$$

Algorithm $(x_1, x_2, \ldots)$:
Step 1 **for** $n = 1, 2, \ldots$ execute Steps 2 through 4.
Step 2 Set $I := \varnothing$; **for** $i := 1, 2, \ldots, n$ execute Step 3.
Step 3 set $\gamma_{i,n} := 1$; L: **if** $\sum_{j=1}^{n} |q_i^n(a_j) - \#a_j(x_1, x_2, ..., x_n)/n| \leq 1/\gamma_{i,n}$
then ($\gamma_{i,n} := \gamma_{i,n} + 1$; **goto** L) **else** $\gamma_{i,n} := \gamma_{i,n} - 1$;
if $\gamma_{i,n} > \gamma_{i,n-1}, \ldots, \gamma_{i,1}$ **then** $I := I \bigcup \{i\}$.
Step 4 **if** $I \neq \varnothing$ **then** $i_n := \min I$ **else** $i_n := i_{n-1}$; output i_n.

Fig. 1. Algorithm 1a

With $q \neq p$ substituted for p in the lefthand side of (A.2), we have that this left-hand side is almost surely unequal 0. Using some probability theory, we can rewrite the Strong Law using only the finite initial segment of the infinite sequence of (copies of) the two-outcome random variables. We use ([6], p 258 ff). For every pair $\epsilon > 0$ and $\delta > 0$, there is an N such that there is a probability $1 - \delta$ or better that for every $r > 0$ all $r + 1$ inequalities:

$$\left| p(a) - \frac{\#a(x_1, x_2, \ldots, x_n)}{n} \right| \leq \epsilon, \tag{A.3}$$

with $n = N, N + 1, ..., N + r$ will be satisfied with probability at least $1 - \delta$. That is, we can say, informally, that with overwhelming probability the left-hand part of (A.3) remains small for all $n \geq N$. Since we deal with all infinite outcomes of i.i.d. draws from the set L according to p, for some sequences that are not pseudo-typical (Remark 4) the inequality (A.3) does not hold. For example, always drawing a_1 while $p(a_1) = \frac{1}{2}$. Therefore, the Strong Law holds "almost surely" and cannot hold "with certainty."

DEFINITION 7: Let k be the least index of an element of $\mathcal{Q}$ such that $q_k = p$. For every i with $1 \leq i < k$, $\max_{a \in L} |q_k(a) - q_i(a)| > 0$ (this follows from the minimality of k). Define

$$\alpha = \min_{1 \leq i < k} \max_{a \in L} |q_k(a) - q_i(a)|.$$

Then, $\alpha > 0$. Let a^i be the a that reaches the maximum in $\max_{a \in L} |q_k(a) - q_i(a)|$ for $1 \leq i < k$, and $\beta = \max_{1 \leq i < k} \{j : a_j \in L \ \& \ a_j = a^i\}$. For every i with $1 \leq i < k$, let t_i be least such that $q_i(a_j) - q_i^t(a_j) \leq \alpha/2$ for every $t \geq t_i$ and $j \leq \beta$. Define τ by $\tau = \max_{1 \leq i < k} t_i$.

The sequence of outputs of the algorithm is $i_1, i_2, \ldots$ such that possibly $i_j < i_{j+1}$, $i_j > i_{j+1}$, or $i_j = i_{j+1}$. Recall that $Q_\infty = \{i : i_n = i \text{ for infinitely many } n\}$.

CLAIM 1: (i) $Q_\infty \neq \varnothing$ almost surely; (ii) if $i \in Q_\infty$ then $i \in Q(p)$ almost surely; (iii) almost surely $\liminf_{n \to \infty} i_n = \min Q(p)$.

Proof: The algorithm outputs a sequence $i_1, i_2, \ldots$. For $n_0 > 3\max\{k, \beta, \tau\} \geq k + \beta + \tau$ the algorithm has considered somewhere along $n = 1, \ldots, n_0$ the approximations $q_1^{t^1}(a^1), \ldots, q_{k-1}^{t^{k-1}}(a^{k-1})$ in Step 3. By Definition 7 these lower approximations are within $\alpha/2$ of the final value of the $q_i(a^i)$ ($1 \leq i < k$). Moreover, again by Definition 7, these final values differ at least by α from the values of $p = q_k$ for all arguments $a^1, \ldots, a^{k-1}$, respectively. Hence the approximations $q_1^t(a^1), \ldots, q_{k-1}^t(a^{k-1})$ differ at least by $\alpha/2$ from $q_k(a^1), \ldots, q_k(a^{k-1})$, respectively, for every $t \geq \tau$. However, Algorithm 1 (Figure 1) does not know k. We have to show how the algorithm handles this information.

Since

$$\lim_{n\to\infty} \left| q_k^t(a) - \frac{\#a(x_1, x_2, \ldots, x_n)}{n} \right| = 0$$

almost surely for every $a \in L$ by (A.2), it follows from the above that

$$\lim_{n\to\infty} \left| q_i^t(a^i) - \frac{\#a^i(x_1, x_2, \ldots, x_n)}{n} \right| \geq \frac{\alpha}{2},$$

almost surely ($1 \leq i < k$). This means that for large enough n we have almost surely that $1/\gamma_{i,n} \geq \alpha/3$ (with $\alpha > 0$ a constant) in Step 3 of the algorithm ($1 \leq i < k$). Then, $\gamma_{i,n} \leq 3/\alpha$ and there is an n_i such that for all $n \geq n_i$ we have $\gamma_{i,n} \leq \gamma_{i,n-1}, \ldots, \gamma_{i,1}$ ($1 \leq i < k$). Thus, for large enough n and $1 \leq i < k$ almost surely we have $i \notin I$ in Step 3.

Almost surely we have $\gamma_{k,n} > \gamma_{k,n-1}, \ldots, \gamma_{k,1}$ for infinitely many n in Step 3 since $\lim_{n\to\infty} 1/\gamma_{k,n} = 0$ almost surely by (A.2). Namely, q_k is the probability mass function according to which $x_1, x_2, \ldots$ is i.i.d. drawn from L. This means that almost surely k is put in I for infinitely many n in Step 3. Moreover, almost surely $i \notin I$ for every $n > n_i$ and $1 \leq i < k$ as we have seen above. Thus, for infinitely many n we have $k = \min I$ and the output in Step 4 is $i_n = k$, almost surely. Therefore $k \in Q_\infty$ almost surely. For $1 \leq i < k$ almost surely $i \notin Q_\infty$ by the above argument. Hence, almost surely $k = \min Q_\infty$ and $k = \liminf_{n\to\infty} i_n$. This shows item (i) and, together with Definition 7, item (iii).

For every $i \geq k$ such that there are infinitely many n for which $\gamma_{i,n} > \gamma_{i,n-1}, \ldots, \gamma_{i,1}$ in Step 3 we have $\lim_{n\to\infty} 1/\gamma_{i,n} = 0$. This again means that q_i is almost surely the probability mass function according to which $x_1, x_2, \ldots$ is i.i.d. drawn from L. Hence $i \in Q(p)$ almost surely. Clearly, i is put in I for infinitely many n. If there are also infinitely many n such that $i = \min I$, then $i \in Q_\infty$. Thus $Q_\infty \subseteq Q(p)$ almost surely. This shows item (ii). ■

■

Proof: OF THEOREM 2. Up to, and exclusive of Definition 7, we follow the proof of Theorem 1.

Algorithm $(x_1, x_2, \ldots)$:
Step 1 **for** $n = 1, 2, \ldots$ execute Steps 2 and 3.
Step 2 Set $I := \varnothing$; **for** $i := 1, 2, \ldots, n$:
if $\sum_{j=1}^{n} |q_i^n(a_j) - \#a_j(x_1, x_2, ..., x_n)/n| \stackrel{+}{=} 0$ **then** $I := I \bigcup \{i\}$.
Step 3 **if** $I \neq \varnothing$ **then** $i_n := \min I$ **else** $i_n := i_{n-1}$; output i_n.

Fig. 2. Algorithm 1b

Algorithm $(x_1, x_2, \ldots)$:
Step 1 **for** $n := 1, 2, \ldots$ execute Step 2.
Step 2 **for** every $a \in L$ occurring in $x_1, x_2, \ldots, x_n$ set $p_n(a) := \#a(x_1, x_2, ..., x_n)/n$.

Fig. 3. Algorithm 2

Since $\lim_{n\to\infty} q_i^n(a_j) = q_i(a_j)$ for every $i, j \geq 1$, by (A.2) almost surely

$$\lim_{n\to\infty} \sum_{j=1}^{n} \left| q_{k'}(a_j)^n - \frac{\#a_j(x_1, x_2, \ldots, x_n)}{n} \right| \stackrel{+}{=} 0,$$

for a k' satisfying $k' \leq k$. Almost surely, the above displayed equation does not hold for i $(1 \leq i < k')$ by similar reasoning as in the proof of Theorem 1. Hence there is an N such that for all $n \geq N$ we have $k' \in I$ and $i \neq I$ for every i $(1 \leq i < k')$ in Step 2 of Algorithm 1b. Consequently, in Step 3 of the algorithm $i_n = k'$ for all $n \geq N$. ■

Proof: OF THEOREM 3.

Algorithm 2 (Figure 3) together with the Strong Law of Large Numbers shows that $\lim_{n\to\infty} p_n(a) = p(a)$ almost surely for every $a \in L$. Here p is the probability mass function of L based on the data sequence $x_1, x_2, \ldots$. Note that in Algorithm 2 the different values of p_n sum to precisely 1 for every $n = 1, 2, \ldots$. ■

Proof: OF THEOREM 4. We need the theory of Kolmogorov complexity [14] (originally in [11] and the prefix version we use here in [13]). A prefix Turing machine is one with a one-way read-only input tape with an distinguished tape cell called the *origin*, a working tape that is a two-way read-write tape on which the computation takes place, and a write-only output tape. At the start of the computation the input tape is infinitely inscribed from the origin onwards, and the input head is on the origin. The machine operates with binary input. If the machine halts then the input head has scanned a segment of the input tape from the origin onwards. We call this initial segment the *program*.

By this definition the set of programs is a prefix code: no program is a proper prefix of any other program. Consider a standard enumeration of all prefix Turing machines T_1, T_2, ... Let U denote a universal Turing machine such that for every $z \in \{0,1\}^*$ and $i \geq 1$ we have $U(i,z) = T_i(z)$. That is, for all finite binary strings z and every machine index $i \geq 1$, we have that U's execution on inputs i and z results in the same output as that obtained by executing T_i on input z. There are infinitely many such U's. Fix one such a U (and with some abuse of notation denote it as U henceforth) and define that conditional *prefix complexity* $K(x|y)$ for all $x, y \in \{0,1\}^*$ by $K(x|y) = \min_p\{|p| : p \in \{0,1\}^* \text{ and } U(p,y) = x\}$. For the same U, define the *time-bounded conditional prefix complexity* $K^t(x|y) = \min_p\{|p| : p \in \{0,1\}^* \text{ and } U(p,y) = x \text{ in } t \text{ steps}\}$. To obtain the unconditional versions of the prefix complexities set $y = \lambda$ where λ is the *empty* word (the word with no letters).

By definition the sets over which the minimum is taken are countable and not empty. It can be shown that $K(x|y)$ is incomputable. Clearly $K^t(x|y)$ is computable if t is computable. Moreover, $K^{t'}(x|y) \leq K^t(x|y)$ for every $t' \geq t$, and $\lim_{t\to\infty} K^t(x|y) = K(x|y)$. Since everything is discrete, there is a least time $t_{x|y} < \infty$ such that $K^{t_{x|y}}(x|y) = K(x|y)$, even though the function $f(x,y)$ defined by $f(x,y) = t_{x|y}$ for all $x, y \in \{0,1\}^*$ may be incomputable.

DEFINITION 8: The language $L = \{a_1, a_2, \ldots, a_m\}$ is finite. We view $a \in L$ as an integer and $a < \infty$. If $x_1, x_2, \ldots, x_n$ is a data sequence with $x_i \in L$ $(1 \leq i \leq n)$, then $K(x_1 \ldots x_n|y) = \min_p\{|p| : p \in \{0,1\}^* \text{ and } U(p,y) = x_{1:n}\}$ where $x_{1:n}$ is an agreed-upon binary encoding of $x_1 x_2 \ldots x_n$. Similarly we define $K^t(x_1 \ldots x_n|y)$.

We now turn to the theory of semicomputable semimeasures. In particular we exhibit a formal criterium that an infinite sequence is "typical" or "random" in Martin-Löf's sense [15].

DEFINITION 9: Let μ be a computable measure on the set of infinite sequences of elements from L. A particular infinite sequence $x = x_1, x_2, \ldots \in L^\infty$ is *typical* or *random* for μ if

$$\sup_n\{\log \frac{1}{\mu(x_1 \ldots x_n)} - K(x_1 \ldots x_n|\mu)\} < \infty,$$

In [14] the definition is different, but is equivalent to the above one by Corollary 4.5.2 there. The measure μ in the conditional of $K(\cdot|\cdot)$ means a finite number of bits that constitute a program that describes μ. Clearly, $K(\mu) < \infty$ since μ is computable. Moreover, according to [14] we have $K(x|y) \geq K(x) - K(y) + O(1)$ for every finite x and y. Therefore, $K(x_1 \ldots x_n) - K(\mu) + O(1) \leq K(x_1 \ldots x_n|\mu) + O(1) \leq$

$K(x_1 \dots x_n) + O(1)$. Hence we can replace the last displayed formula by

$$\sup_n \{\log \frac{1}{\mu(x_1 \dots x_n)} - K(x_1 \dots x_n)\} < \infty. \tag{A.4}$$

Our data is, by assumption, typical (equivalently random) for some computable measure μ. That is, the data $x_1, x_2, \dots$ satisfies (A.4) with respect to μ. We can effectively enumerate all and only semimeasures that are semicomputable from below as the elements listed in $\mathcal{M}$ of (IV.2).

REMARK 6: We stress that the data is possibly μ-random and μ'-random for different measures μ and μ'. In general it can be so for many measures in $\mathcal{M}$. Therefore we cannot speak of the true measure, but only of a measure for which the data is typical. $\diamond$

To eliminate the undesirable lower semicomputable semimeasures among $\mu_1, \mu_2 \dots$ we sieve out the ones that are not measures, and among the measures the ones that do not show $x_1, x_2, \dots$ random to it. To do so, we conduct for elements of $\mathcal{M}$ a test for both properties. Since the test is computational we need the (in)equality relations $\stackrel{+}{\leq}$, $\stackrel{+}{\geq}$, $\stackrel{\pm}{=}$, $\stackrel{+}{<}$, $\stackrel{+}{>}$ of Definition 4. In particular this is needed in the properties in Claim 3. These properties are used in Step 3 of Algorithm 3 (Figure 4).

DEFINITION 10: Since the algorithms have a round-off error, we can test only $\stackrel{\pm}{=} 0$ or not $\stackrel{\pm}{=} 0$. Consequently, we count semimeasures as measures if they satisfy (IV.1) but deviate from equalities by a very small additive term only. More precisely, μ_i in the list $\mathcal{M}$ is counted as a *semimeasure but not as a measure* if $\mu_i(z) - \sum_{a \in L} \mu_i(za) \stackrel{+}{>} 0$. If $\mu_i(z) - \sum_{a \in L} \mu_i(za) \stackrel{\pm}{=} 0$ then we view μ_i as a *measure*.

CLAIM 2: Assume that μ_i in $\mathcal{M}$ is a semimeasure but not a measure according to Definition 10. Then, there is a least $z \in L^*$ and a least n_i such that $\mu_i^n(z) - \sum_{a \in L} \mu_i^n(za) \stackrel{+}{>} 0$ for every $n \geq n_i$.

Proof: Since μ_i is lower semicomputable and $|L| < \infty$, for every $z \in L^*$ there is an n_z such that for every $n \geq n_z$ we have

$$0 \leq \mu_i(z) - \mu_i^n(z) \stackrel{+}{=} 0$$

and for every $a \in L$ we have

$$0 \leq \sum_{a \in L} \mu_i(za) - \sum_{a \in L} \mu_i^n(za) \stackrel{+}{=} 0.$$

Therefore,

$$\mu_i^n(z) - \sum_{a \in L} \mu_i^n(za) \stackrel{+}{=} \mu_i(z) - \sum_{a \in L} \mu_i(za) \stackrel{+}{>} 0. \tag{A.5}$$

■

DEFINITION 11: Let μ_i be an element of $\mathcal{M}$. Assume that for all n we have $i, j > 0$ and $i +$

$j = n$. Define $Z_{i,j,n} = \{z \in L^* : \ |z| \leq j, \ \mu_i^n(z) - \sum_{a\in L} \mu_i^n(za) \stackrel{+}{>} 0\}$ and $\Delta_{i,n} = \max\{\Delta : Z_{i,j-\Delta,n-\Delta} \bigcap \cdots \bigcap Z_{i,j,n} \neq \varnothing\}$.

REMARK 7: The set $Z_{i,j,n}$ contains all strings $z \in L^*$ of at least length j such that $i + j = n$ and $\mu_i^n(z) - \sum_{a\in L} \mu_i^n(za) \stackrel{+}{>} 0$. The intersection $Z_{i,j-\Delta,n-\Delta} \bigcap \cdots \bigcap Z_{i,j,n}$ is the set of all strings z of length at least $j - \Delta$ that witness that the approximations $\mu_i^{n-\Delta}, \ldots, \mu_i^n$ are all semimesures but not measures. The quantity $\Delta_{i,n}$ is the maximum number of approximations before and including the nth approximation of μ_i such that the same $z \in L^*$ of length at least $j - \Delta_{i,n}$ with $i + j = n$ witnesses that all these approximations are semimeasure but not measures: $\mu_i^{n'}(z) \stackrel{+}{>} \sum_{a\in L} \mu_i^{n'}(za)$ for every n' such that $n - \Delta_{i,n} \leq n' \leq n$. $\diamond$

CLAIM 3: Let μ_i be an element of the list $\mathcal{M}$ and n_i be as in Claim 2. If μ_i is not a measure according to Definition 10, then for every $n \geq n_i$ we have $\Delta_{i,n} \geq n - n_i$ for all $n \geq n_i$ and $\Delta_{i,n} = \Delta_{i,n-1} + 1$ for all $n > n_i$. If μ_i is a measure according to Definition 10, then for every n there is a greatest $c < \infty$ such that $\Delta_{i,n} < n - c$ and c goes to ∞ with growing n. We have $\Delta_{i,n} < \max\{\Delta(i, n-1), \ldots, \Delta(i, 1)\} + 1$ for infinitely many n iff μ_i is a measure.

Proof: If μ_i is not a measure then by Claim 2 there is a $z \in L^*$ such that $\mu_i^n(z) - \sum_{a\in L} \mu_i^n(za) \stackrel{+}{>} 0$ for all $n \geq n_i$. This z will never leave the sets $Z_{i,j,n}$ ($|z| \leq j = n-i$) for $n \geq n_i$. Therefore, $\Delta_{i,n} \geq n - n_i$ and $\Delta_{i,n} = \Delta_{i,n-1} + 1$ for all $n > n_i$.

If μ_i is a measure then $\lim_{n\to\infty}(\mu_i^n(z) - \sum_{a\in L} \mu_i^n(za)) = 0$. Therefore, for every z there is a least $n_{i,z} < \infty$ such that $\mu_i^n(z) - \sum_{a\in L} \mu_i^n(za) \stackrel{+}{=} 0$ for all $n \geq n_{i,z}$. Hence, for every j and every $z \in L^*$ with $|z| \leq j$ we have $z \notin Z_{i,j,n}$ for all $n \geq n_{i,z}$. Thus, every finite string in $Z_{i,j,n}$ is not a member of $Z_{i,j+n'-n,n'}$ any more for every $n' > n$. Therefore, for every n there is a greatest $c < \infty$ such that $\Delta_{i,n} < n - c$ and c goes to ∞ with growing n. This implies that

$$\Delta_{i,n} < \max\{\Delta(i, n-1), \ldots, \Delta(i, 1)\} + 1$$

for infinitely many n. In view of the above property of semimeasures that are not measures according to Definition 10, μ_i is a measure iff the last displayed equation holds. ■

REMARK 8: To make everything effective (computable) for Algorithm 3 (Figure 4) we do not use prefix complexity as in (A.4) but the time-bounded analog as defined. By using dovetailing, that is, $n = 1, 2, \ldots$ with all combinations of $i, j > 0$ such that $i + j = n$ and n is the number of steps, as in Definition 11, with growing n every $\mu_i^n(z)$ with $|z| \leq j$ for every particular i, j, n is computed and considered.

In Algorithm 3 one wants to determine the indexes i of elements μ_i in the list $\mathcal{M}$ such that μ_i is a measure. This happens in Steps 2 and 3 as follows. For growing n the fact that index i is selected to go in I means that μ_i is possibly a measure. Eventually, if n is large anough this possibility will turn into a certainty. Moreover, this will hold for every measure μ_i. Thus, for every i with growing n, but not computably, it is decided that μ_i is a measure or not. If it is a measure, then it keeps on figuring in the second part of the algorithm, if it is decided not to be a measure then it will not figure in the second part of the algorithm.

This second part of the algorithm determines for which measures the data $x_1, x_2, \ldots$ is random. Note that this part initially also may consider nonmeasures. But with growing n, because of the first part of the algorithm, for every index i it will consider only measures but not nonmeasures μ_i. If for some n the index i is selected then in Step 5 the algorithm computes the n-approximations $\rho(i, j', n) := \log 1/\mu_i^n(x_1 \ldots x_{j'}) - K^n(x_1 \ldots x_{j'})$ ($1 \leq j' \leq j$, $j = n - i$). This in order to obtain approximations to the elements constituting the initial segment of the sequence of which equation (A.4) takes the supremum. In Step 6 the algorithm takes the maximum $\sigma(i, n, j') := \max\{\rho(i, j'', n) : 1 \leq j'' \leq j'\}$ over the initial segments of this initial segment. In Step 7 it determines for every μ_i concerned how long the longest flat plateau of this sequence of maxima is.

Suppose μ_i is a measure for which $x_1, x_2, \ldots$ is random. If there exists a n_0 such that the n_0-approximation of $\rho(i, j_0, n_0)$ has reached the supremum in (A.4), then for every $j \geq j_0$ and $n \geq n_0$ we have that $\rho(i, j, n)$ reaches this supremum. (Reaching means "is within a unit.") To select a μ_i one looks for the $\rho(i, \cdot, \cdot)$ that gives the longest flat plateau, that is, has reached the supremum in (A.4) the soonest in terms of the initial segment of $x_1, x_2, \ldots$. Thus, in Step 7 the algorithm compares the length of the flat plateau with the top score, and changes the latter if it is exceeded. In Step 8 the algorithm either selects the index resulting in a new or equal top score or goes with the index of approximation $n - 1$. Note that with growing n nonmeasures are excluded in the first part of the algorithm. Thus, with growing n, the measure μ_i that reaches (A.4) soonest and has the longest flat plateau, has an index i that is not (eventually) excluded by the first part of the algorithm. $\diamond$

DEFINITION 12: Let μ_i in list $\mathcal{M}$ be a measure according to Definition 10 and $x_1, x_2, \ldots$ be an infinite sequence of elements from L. By (A.4) we have $\mu_i \in \mathcal{M}(x)$ iff there is a $\sigma_i < \infty$ such that

$$\sigma_i = \max_j \left\{ \left\lfloor \log \frac{1}{\mu_i(x_1 \ldots x_j)} - K(x_1 \ldots x_j) \right\rfloor - 1 \right\} \tag{A.6}$$

for $1 \leq j \leq \infty$. Define m_i as the least j for which σ_i is reached.

Algorithm $(x_1, x_2, \ldots)$:

Step 1 $m := 0$; **for** $n = 1, 2, \ldots$ execute Steps 2 through 8.

Step 2 $I := \varnothing$; **for** every $i, j > 0$ satisfying $i + j = n$, compute $\Delta_{i,n}$ (note $j = n - i$) and execute Steps 3 through 7.

Step 3 **if** $\Delta_{i,n} < \max\{\Delta(i, n-1), \ldots, \Delta(i, 1)\} + 1$ **then** $I := I \bigcup \{i\}$ (by Claim 3, $i \in I$ for infinitely many n iff μ_i is a measure).

Step 4 **if** $I \neq \varnothing$ **then** execute Steps 5 through 7 for every $i \in I$.

Step 5 **for** $j' := 1, \ldots, j$ set $\rho(i, j', n) := \log 1/\mu_i^n(x_1 \ldots x_{j'}) - K^n(x_1 \ldots x_{j'})$.

Step 6 **for** $j' := 1, \ldots, j$ set $\sigma(i, n, j') := \max\{\rho(i, j'', n) : 1 \leq j'' \leq j'\}$.

Step 7 $s(i, n) := \max\{s : \lfloor \sigma(i, n, r) \rfloor - 1 = \cdots = \lfloor \sigma(i, n, r + s) \rfloor - 1,\ 1 \leq r \leq r + s \leq j\}$; **if** $s(i, n) \leq m$ **then** $I := \varnothing$ **else** $m := s(i, n)$.

Step 8 **if** $I \neq \varnothing$ **then** $i_n := \min\{i : s(i, n) = m\}$ **else** $i_n := i_{n-1}$; output i_n.

Fig. 4. Algorithm 3

REMARK 9: In Definition 12 we have replaced "measure" by "measure according to Definition 10." We have replaced the "$\sup$" in (A.4) by "$\max$" by rounding down. Moreover, by rounding down and subtracting 1 we have taken care that $m_i < \infty$. $\diamond$

(Proof of the theorem continued.) The sequence of outputs of Algorithm 3 are indexes $i_1, i_2, \ldots$ of lower semicomputable semimeasures in $\mathcal{M}$, such that possibly $i_j < i_{j+1}$, $i_j > i_{j+1}$, or $i_j = i_{j+1}$. By Claim 3, for every measure (Definition 10) μ_i in $\mathcal{M}$ we have $i \in I$ in Step 3 of the algorithm for infinitely many n. For large enough n index $i \notin I$ of a nonmeasure μ_i. So for every index i there is a large anough n such that $i \in I$ only if μ_i is a measure. These measures are treated in Steps 4 through 8. By assumption the data $x = x_1, x_2, \ldots$ is random (typical) for some measure in $\mathcal{M}$. Let this be measure μ_k.

Let us look at long plateaus $s(i, n)$ for measures μ_i such that either $x = x_1, x_2, \ldots$ is not random to it or $m_i > n$ (with m_i as in Definition 12). For a measure μ_i such that $x = x_1, x_2, \ldots$ is not random to it, $s(i, n)$ can be any constant c. However the lefthand side of (A.4) goes to infinity in this case, so we know that $s(i, n') = 1$ for some $n' \geq n$. Since the data $x = x_1, x_2, \ldots$ is random to μ_k, for all n that are large enough $s(k, n) = s(k, n-1) + 1$ by Claim 3. Hence, for large enough n we have $s(k, n) > s(i, n)$, since $s(k, n) \to \infty$ with $n \to \infty$. For a measure μ_j with $j \neq k$ such that $x = x_1, x_2, \ldots$ is random to it, $s(j, n)$ can be a constant $c < \sigma_j$ while $n < m_j$. That is, the maximum in (A.6) has not yet been reached. Again we know that $s(j, n') = 1$ for some $n' > n$. Hence without loss of generality we can exclude cases like measures μ_i and μ_j. Let us consider only measures μ_l and steps n, such that $x = x_1, x_2, \ldots$ is random to μ_l and $n \geq m_l$.

Without loss of generality we can assume that we compute $s(i,n)$ for every measure μ_i in $\mathcal{M}$ and every n according to Steps 6 and 7, and not just when $i \in I$. In particular we can do so for μ_k. By Claim 3, for every n that is large enough we have $s(k,n) \leq s(i,k-1)+1$ and $s(k,n) = s(k,n-1)+1$. Let i satisfy $m_k < i$. Then, $s(i,m_k) = 0$ (the set over which $s(i,m_k)$ is maximized in Step 7 equals $\varnothing$). Since $s(i,n) \leq s(i,n)+1$ for all n, we have $s(i,n) < s(k,n) \leq m$ (with m as in Step 7) for all $n \geq m_k$. Hence $i_n \neq i$ in Step 8 if n is large enough. Let A be the set of measures μ_i with $i < m_k$. Then $|A| < m_k$ and $M_\infty \subseteq A$. Hence $|M_\infty| < m_k < \infty$.

Since $|M_\infty| < m_k$, and by Steps 5,6,7 the output of Algorithm 3 is an infinite sequence $i_1, i_2, \ldots$, we have that some $i_0 < m_k$ occurs infinitely often in this sequence. Hence $M_\infty \neq \varnothing$.

Let the index i of a measure μ_i occur in M_∞. Then $s(i,n)$ is larger than m in Step 7 for infinitely many n. (It is impossible that $i_n = i_{n-1}$ for an infinitely long run of n's since $m \to \infty$ with $n \to \infty$. The latter statement is a consequence of $s(k,n)$ growing with n.) Since $m \to \infty$ with $n \to \infty$ we have $s(i,n) \to \infty$ with $n \to \infty$. By the definition of $s(\cdot,\cdot)$ and (A.6), the data $x = x_1, x_2, \ldots$ is random with respect to the measure μ_i. Hence, $M_\infty \subseteq M(x)$. This proves the theorem. ■

REFERENCES

[1] D. Angluin, Identifying languages from stochastic examples, Yale University, Dept. of Computer Science, Technical report, New Haven, Conn., USA, 1988.

[2] S.B. Cooper, P. Odifreddi (2003). Incomputability in nature, Pp. 137–160 in S. B. Cooper and S. S. Goncharov, Eds., *Computability and Models: Perspectives East and West*, Plenum, New York, 2003.

[3] T.M. Cover and J.A. Thomas, *Elements of Information Theory*, Wiley, New York, 1991.

[4] E.M. Gold, Limiting recursion, *J. Symb. Logic*, 30(1965), 28-48.

[5] E.M. Gold, Language identification in the limit, *Inform. Contr.*, 10(1967), 447–474.

[6] W. Feller, *An Introduction to Probability Theory and Its Applications*, Vol. 1, Wiley, New York, 1968 (third edition).

[7] A. Hsu, N. Chater and P.M.B. Vitányi, The probabilistic analysis of language acquisition: Theoretical, computational, and experimental analysis, *Cognition*,120(2011), 380–390.

[8] S. Jain, D.N. Osherson, J.S. Royer, A. Sharma, *Systems that Learn*, MIT Press, Cambridge, Mass., 1999 (second edition).

[9] A.N. Kolmogorov, *Grundbegriffe der Wahrscheinlichkeitsrechnung*, Springer-Verlag, Berlin, 1933.

[10] A.N. Kolmogorov, Sur la loi forte des grandes nombres, *C. r. Acad. Sci. Paris*, 191(1930), 910–912. See also A.N. Kolmogorov, *Grundbegriffe der Wahrscheinlichkeitsrechnung*, Springer-Verlag, Berlin, 1933. See also F.P. Cantelli, Sulla probabilitá come limite della frequenza, *Rendiconti della R. Academia dei Lincei, Classe di scienze fisische matematiche e naturale, Serie* 5^a, 26(1917), 39–45.

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

[12] M. Kracht, *The Mathematics of Language*, Mouton & de Gruyter, Berlin, 2003.

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

[14] M. Li and P.M.B. Vitányi, *An Introduction to Kolmogorov Complexity and Its Applications*, Springer-Verlag, New York, 2008 (third edition).

[15] P. Martin-Löf, The definition of random sequences, *Inform. Control*, 9:6(1966), 602–619.

[16] S. Pinker, Formal models of language learning, *Cognition* 7(1979), 217–283.

[17] K.R. Popper, *The Logic of Scientific Discovery*, Hutchinson, London, 1959.

[18] Z.W. Pylyshyn, Z. W., *Computation and Cognition*, MIT Press, Cambridge, Mass., 1984.

[19] F. Rieke, D. Warland, R. de Ruyter van Steveninck, W. Bialek, (1997). *Spikes: Exploring the Neural Code*, MIT Press, Cambridge, Mass., 1997.

[20] A.M. Turing, On computable numbers, with an application to the Entscheidungsproblem, *Proc. London Mathematical Society* 2, 42(1936), 230–265, ”Correction”, 43(1937), 544–546.

[21] A.K. Zvonkin and L.A. Levin, The complexity of finite objects and the development of the concepts of information and randomness by means of the theory of algorithms, *Russian Math. Surveys*, 25:6(1070), 83–124.