Algorithmic identification of probabilities is hard

Laurent Bienvenu[a,*], Santiago Figueira[b], Benoit Monin[c], Alexander Shen[a]

[a] *LIRMM, CNRS & Université de Montpellier, 161 rue Ada, 34095 Montpellier Cedex 5, France*

[b] *Universidad de Buenos Aires and CONICET, Pabellón I - Ciudad Universitaria, Buenos Aires, Argentina*

[c] *LACL, Université Paris 12, 61 avenue du Général de Gaulle 94010 Créteil Cedex, France*

Abstract

Suppose that we are given an infinite binary sequence which is random for a Bernoulli measure of parameter p. By the law of large numbers, the frequency of zeros in the sequence tends to p, and thus we can get better and better approximations of p as we read the sequence. We study in this paper a similar question, but from the viewpoint of inductive inference. We suppose now that p is a computable real, and one asks for more: as we are reading more and more bits of our random sequence, we have to eventually guess the exact parameter p (in the form of its Turing code). Can one do such a thing uniformly for all sequences that are random for computable Bernoulli measures, or even for a 'large enough' fraction of them? In this paper, we give a negative answer to this question. In fact, we prove a very general negative result which extends far beyond the class of Bernoulli measures. We do however provide a weak positive result, by showing that looking at a sequence X generated according to some computable probability measure, we can eventually guess a sequence of measures with respect to which X is random in Martin-Löf's sense.

Keywords: Algorithmic learning theory, algorithmic randomness.
2010 MSC: 03D32, 68Q30, 68Q32

*Corresponding author

1. Introduction

The study of learnability of computable sequences is concerned with the following problem. Suppose we have a black box that generates some infinite computable sequence of bits $X = X(0)X(1)X(2), \ldots$ We do not know the program running in the box, and want to guess it by looking at finite prefixes

$$X\restriction n = X(0) \ldots X(n-1)$$

for increasing n. There could be different programs that produce the same sequence, and it is enough to guess one of them (since there is no way to distinguish between them by just looking at the output bits). The more bits we see, the more information we have about the sequence. For example, it is hard to say something about a sequence seeing only that its first bit is a 1, but looking at the prefix

$$11001001000011111011010101000$$

one may observe that this is a prefix of the binary expansion of π, and guess that the machine inside the box does exactly that (though the machine may as well produce the binary expansion of, say, 47627751/15160384).

The hope is that, as we gain access to more and more bits, we will *eventually* figure out how the sequence X is generated. More precisely, we hope to have a computable function $\mathfrak{A}$ such that for every computable X, the sequence

$$\mathfrak{A}(X\restriction 1),\ \mathfrak{A}(X\restriction 2),\ \mathfrak{A}(X\restriction 3), \ldots$$

converges to a program (= Turing machine) that computes X. This is referred to as *identification in the limit*, and can be understood in two ways:

- Strong success: for every computable X, the above sequence converges to a single program that produces X.
- Weak success: for every computable X, all but finitely many terms of the above sequence are programs that produce X (but these programs may be different).

The first type of success is often referred to as *explanatory* (EX) and the second type as *behaviorally correct* (BC). Either way, such an algorithm $\mathfrak{A}$ does not exist in general. The main obstacle is that certain machines are not total (produce only finitely many bits), and distinguishing total machines

from non-total ones cannot be done computably. (If we restrict ourselves to some decidable class of total machines, e.g., primitive recursive functions, then exact learning is possible: let $\mathfrak{A}(u)$ be the first machine in the class that is compatible with u.) We refer the reader to [11] for a detailed survey of learnability of computable functions.

Recently, Vitányi and Chater [9] proposed to study a related problem. Suppose that instead of a total deterministic machine, the black box contains an *almost total probabilistic machine* M. By "almost total" machine we mean a randomized algorithm that produces an infinite sequence with probability 1. The output distribution of such a machine is a computable probability measure μ_M over the space 2^ω of infinite binary sequences. Again, our ultimate goal is to guess what machine is in the box, i.e., to give a reasonable explanation for the observed sequence X. For example, observing the sequence

$$0001111111100001100000000011111111111111$$

one could guess that M is a probabilistic machine that starts with 0 and then chooses each output bit to be equal to the previous one with probability $4/5$ (so the change happens with probability $1/5$), making all the choices independently.

What should count as a good guess for some observed sequence? Again there is no hope to distinguish between some machine M and another machine M' that has the same output distribution $\mu_{M'} = \mu_M$. So our goal should be to reconstruct the output distribution and not the specific machine.

But even this is too much to ask for. Assume that we have agreed that some machine M is a plausible explanation for some sequence X. Consider another machine M' that starts by tossing a coin and then (depending on the outcome) either generates an infinite sequence of zeros or simulates M'. If X is a plausible output of M, then X is also a plausible output for M', because it may happen (with probability $1/2$) that M' simulates M.

A reasonable formalization of 'good guess' is provided by the theory of algorithmic randomness. As Chater and Vitányi recall, there is a widely accepted formalization of "plausible outputs" for an almost total probabilistic machine with output distribution μ: the notion of Martin-Löf random sequences with respect to μ. These are the sequences which pass all effective statistical tests for the measure μ, also known as μ-Martin-Löf tests. (We assume that the reader is familiar with algorithmic randomness and

Kolmogorov complexity. The most useful references for our purposes are [5] and [8]). Having this notion in mind, one could look for an algorithm $\mathfrak{A}$ with the following property:

> *For every almost total probabilistic machine M with output distribution μ_M, for μ_M-almost all X, the sequence*
>
> $$\mathfrak{A}(X\restriction 1), \mathfrak{A}(X\restriction 2), \mathfrak{A}(X\restriction 3), \ldots$$
>
> *identifies in the limit an almost total probabilistic machine M' such that X is $\mu_{M'}$-Martin-Löf random.*

Note that this requirement uses two machines M and M' (more precisely, their output distributions): the first one is used when we speak about "almost all" X, and the second is used in the definition of Martin-Löf randomness. Here M' may differ from M and, moreover, may be different for different X.

In an early version of [9], Vitányi and Chater claimed that this can be done in the strongest sense (EX): the guesses $\mathfrak{A}(X\restriction n)$ converge to a single code of some machine M'. It is this claim which motivated the present paper, and we in fact show that this is incorrect, and that even a much weaker goal cannot be achieved. This was corrected in the more recent versions of [9].

Let us consider a rather weak notion of success: $\mathfrak{A}$ *succeeds* on X if there exists $c > 0$ such that for all sufficiently large n the guess $\mathfrak{A}(X\restriction n)$ is a machine M' such that X is $\mu_{M'}$-Martin-Löf random with randomness deficiency less than c with respect to measure $\mu_{M'}$. (Here we use the notion of uniform randomness deficiency $\mathbf{d}(X\,|\,\mu)$ where X is a sequence and X is a measure, see [1] for the details.) So the machines $\mathfrak{A}(X\restriction n)$ may be different, we only require that X is Martin-Löf random (with bounded deficiency) for almost all of them. (If almost all machines $\mathfrak{A}(X\restriction n)$ generate the same distribution and X is Martin-Löf random with respect to this distribution, this condition is guaranteed to be true; note that the second argument in $\mathbf{d}(X\,|\,\mu)$ is the measure, not the machine that generates it.)

Moreover, we require $\mathfrak{A}$ to be successful only with some positive probability instead of probability 1, and only for machines from some class: for every machine M from this class of machines, $\mathfrak{A}$ is required to succeed with μ_M-probability at least $\delta > 0$, for some δ independent of M.

Our first main result is that even with these easier requirements, there cannot be such an algorithm $\mathfrak{A}$ (as long as the class of machines is not too narrow: if for example it contains only one machine M, the algorithm $\mathfrak{A}$

can always produce a code for this machine). This was the result proven by Bienvenu, Monin and Shen in an earlier version of this paper presented at the conference ALT 2014 [3]. The material of this earlier version is presented in Section 3, but we give a much more natural proof of the main result.

At the time of writing [3], the authors conjectured that the requirement of bounded deficiency was unnecessary but could not find a proof. In joint work with Figueira, they proved that, surprisingly, the opposite is true: without the bounded deficiency assumption, there *does* exist an algorithm $\mathfrak{A}$ satisfying all our requirements. This is the second main result of the present paper, which is presented in Section 4.

2. Background and notation

We denote by 2^ω the set of infinite binary sequences and by $2^{<\omega}$ the set of finite binary sequences (or *strings*). The length of a string σ is denoted by $|\sigma|$. The empty string (string of length 0) is denoted by Λ. For σ, τ strings, we write $\sigma \preceq \tau$ if σ is a prefix of τ. The n-th element of a sequence $X(0)X(1)\ldots$ is the value $X(n-1)$ (assuming that the length of X is at least n); the string $X\restriction n = X(0)X(1)\ldots X(n-1)$ is *n-bit prefix of X*. We write $\sigma \preceq X$ if the string σ is a prefix of the infinite sequence X (i.e., $X\restriction|\sigma| = \sigma$). The space 2^ω is endowed with the distance d defined by

$$d(X,Y) = 2^{-\min\{n : X(n) \neq Y(n)\}}.$$

This distance is compatible with the product topology generated by *cylinders*

$$[\sigma] = \{X \in 2^\omega \ : \ \sigma \preceq X\}.$$

A cylinder is both open and closed (= *clopen*). Thus, any finite union of cylinders is also clopen. It is easy to see, by compactness, that the converse holds: every clopen subset of 2^ω is a finite union of cylinders. We say that a clopen set C has *granularity at most n* if it can be written as a finite union of cylinders $[\sigma]$ with all σ's of length at most n. We denote by Γ_n the family of clopen sets of granularity at most n.

We now give a brief review of the 'computable analysis' aspects of the space of probability measures. For a more thorough exposition of the subject, the main reference is [5].

The space of Borel probability measures over 2^ω is denoted by $\mathcal{P}$. In the rest of the paper, when we talk about a 'measure', we mean an element of the

space $\mathcal{P}$. This space is equipped with the weak topology. Several classical distances are compatible with this topology; for example, one may use the distance ρ constructed as follows. For $\mu, \nu \in \mathcal{P}$, let $\rho_n(\mu, \nu)$ (for an integer n) be the quantity

$$\rho_n(\mu, \nu) = \max_{C \in \Gamma_n} |\mu(C) - \nu(C)|$$

and then set

$$\rho(\mu, \nu) = \sum_n 2^{-n} \rho_n(\mu, \nu).$$

The *open* (resp. *closed*) *ball* $\mathcal{B}$ *of center* μ *and radius* r is the set of measures ν such that $\rho(\mu, \nu) < r$ (resp. $\rho(\mu, \nu) \leq r$). In the space of measures, the closure $\overline{\mathcal{B}}$ of the open (or closed) ball $\mathcal{B}$ of center μ and radius r is the closed ball of center μ and radius r.

This distance makes $\mathcal{P}$ a computable metric space. Thus, one can define partial computable functions from some discrete space $\mathcal{D}$ (such as $\mathbb{N}$) to $\mathcal{P}$ via type-2 computability as follows. Consider an algorithm G which for every input $z \in \mathcal{D}$ enumerates a (finite or infinite) list of rational balls[1] $\mathcal{B}_1$, $\mathcal{B}_2$,... in $\mathcal{P}$ such that $\overline{\mathcal{B}}_{i+1} \subseteq \mathcal{B}_i$, and the radius of $\mathcal{B}_i$ is less than 2^{-i}. For a given z, the list of balls can be finite or infinite. Given an algorithm G, we consider a partial function F from $\mathcal{D}$ to $\mathcal{P}$; its domain consists of points z such that $G(z)$ enumerates an infinite sequence of balls, and $F(z)$ is the (unique) common point of these balls. Functions obtained in this way are called (type-2) computable partial functions from $\mathcal{D}$ to $\mathcal{P}$. For convenience, we often identify F to its enumeration algorithm G.

The *randomness deficiency* function $\mathbf{d}$ is the largest, up to additive constant, function $f : 2^\omega \times \mathcal{P} \to \mathbb{N} \cup \{\infty\}$ such that

- f is lower semi-computable (i.e., $f^{-1}((k, \infty])$ is an effectively open subset of the product space $2^\omega \times \mathcal{P}$, uniformly in k)
- for every $\mu \in \mathcal{P}$, for every integer k, $\mu\{X : f(X, \mu) > k\} < 2^{-k}$

We use the usual notation $\mathbf{d}(X|\mu)$ instead of $\mathbf{d}(X, \mu)$. We say that X is (Martin-Löf) random relative to measure μ if $\mathbf{d}(X|\mu) < \infty$.

[1] We fix some natural dense set of rational-valued finitely representable measures. Rational balls are balls of rational radius with centers in this set. Such balls can be finitely represented. The family of rational balls is a base of the topology on $\mathcal{P}$.

If $\mathcal{B}$ is a rational ball (open or closed), we write $\mathbf{d}(X|\mathcal{B}) = \inf_{\mu \in \overline{\mathcal{B}}} \mathbf{d}(X|\mu)$. By effective compactness of the space of measures, one can verify that this quantity is also lower-semicomputable uniformly in X and a code for $\mathcal{B}$. Finally, if a partial function F from $\mathcal{D}$ to $\mathcal{P}$ is computable, we write $\mathbf{d}(X|F(z))$ for the supremum of $\mathbf{d}(X|\mathcal{B})$ over balls $\mathcal{B}$ enumerated by the algorithm computing $F(z)$. This notation is consistent in the sense that if $F(z)$ is defined and μ is its value, then $\mathbf{d}(X|F(z)) = \mathbf{d}(X|\mu)$; if $F(z)$ is undefined, the value of $\mathbf{d}(X|F(z))$ is determined by the last (smallest) ball in the enumeration.

At this point the way we view measures – as points of the space $\mathcal{P}$ – does not match the presentation of the introduction, where we asked the learning algorithm to guess, on input X, a sequence of probabilistic machines M_i such that for almost all i, M_i is almost total and X is a plausible output for M_i. The reason is that in fact there are three ways one can think of measures, which are equivalent for our purposes:

(a) A measure is a point of $\mathcal{P}$.

(b) By Caratheodory's theorem, a measure μ can be identified with the function $\sigma \mapsto \mu([\sigma])$: given any function $f : 2^{<\omega} \to [0,1]$ such that $f(\Lambda) = 1$ and $f(\sigma 0) + f(\sigma 1) = f(\sigma)$, there is a unique measure μ such that $\mu([\sigma]) = f(\sigma)$ for all σ. For example, the uniform measure λ is the unique measure such that $\lambda([\sigma]) = 2^{-|\sigma|}$ for all σ, and the Bernoulli measure β_p of parameter $p \in [0,1]$ is the unique measure satisfying $\beta_p([\sigma 1]) = p \cdot \beta_p([\sigma])$ for all σ.

(c) Consider a Turing functional M, which one might think of as a Turing machine with a read-only input tape and a write-only output tape. We say that M is defined on X if M prints an infinite sequence Y given X on the input tape. When M is defined on λ-almost every X, we say that M is *almost total* and then the function

$$\mu_M(\sigma) = \lambda\{X : M(X) \succeq \sigma\}$$

defines a measure in the sense of item (b). The machine M can then be regarded as a probabilistic machine where X is the random seed and the output is a random variable whose distribution is μ_M.

These approaches are equivalent both in the classical and effective realm, as is well known. For our purposes it is important that a learning algorithm

$\mathfrak{A}$ can be transformed into a computable partial function from strings to measures that is defined on some string x if and only if $\mathfrak{A}(x)$ is an almost total machine. (We produce smaller and smaller balls as soon as the machine $\mathfrak{A}(x)$ is more and more total, so to say.) This allows us to translate the initial statement about learning the machines into the language of computable functions used in the proof of Theorem 3.2 (Section 3) below. The second approach (b) will be convenient for the positive result (Theorem 4.1, Section 4).

We end this introduction with a discussion on a concept we will need to state the main theorem of Section 3: orthogonality. Two measures $\mu, \nu \in \mathcal{P}$ are said to be *orthogonal* if there is a set $\mathcal{X} \subseteq 2^\omega$ such that $\mu(\mathcal{X}) = 1$ and $\nu(\mathcal{X}) = 0$ (taking at the complement of $\mathcal{X}$, we see that this is a symmetric relation). This is equivalent to asking that for each ε, there is a set $\mathcal{X}_\varepsilon$ such that $\mu(\mathcal{X}_\varepsilon) \geq 1 - \varepsilon$ and $\nu(\mathcal{X}_\varepsilon) < \varepsilon$ (indeed one can then take $\mathcal{X} = \bigcap_i \bigcup_j \mathcal{X}_{2^{-i-j}}$).

The class of Bernoulli measures provides an easy example of orthogonality: if $p \neq q$, the Bernoulli measures β_p and β_q (see definition above) are orthogonal (by the law of large numbers, taking for $\mathcal{X}$ the set of sequences with a limit frequency of zeroes equal to p, we have $\beta_p(\mathcal{X}) = 1$ and $\beta_q(\mathcal{X}) = 0$).

The important fact we need is that when two computable measures μ and ν are orthogonal, they share no random element, i.e, $\mathbf{d}(X|\mu)$ and $\mathbf{d}(X|\nu)$ cannot both be finite. For a proof of this result, see for example [2].

3. Identifying measures is hard

Now we are ready to give a precise formulation of our main negative result. The *learning algorithm* is a partial computable function $\mathfrak{A}$ from $2^{<\omega}$ to $\mathcal{P}$; it gets the prefix $X\restriction n$ of a sequence X and computes (in type-2 sense) some measure $\mathfrak{A}(X\restriction n)$. We say that $\mathfrak{A}$ *BC-succeeds* on a sequence $X \in 2^\omega$ if $\mathfrak{A}(X\restriction n)$ is defined and outputs the same computable measure μ for all sufficiently large n, and X is Martin-Löf random with respect to this measure μ. This is a weaker requirement than exact (EX) success mentioned above: the algorithm is obliged to produce the same measure (for almost all n), but is not obliged to produce the same machine (or the sequence of balls converging to this measure). Our main result, in its weak form, says that this goal cannot be achieved for all sequences that are random with respect to some computable measure:

Theorem 3.1. *No algorithm $\mathfrak{A}$ can BC-succeed on every sequence X that is random with respect to some computable measure.*

As we have discussed, we prove a stronger version of this result—stronger in three directions.

First, we require the learning algorithm to succeed only on sequences that are random with respect to measures in some restricted class, for example, the class of Bernoulli measures (the main particular case considered in the original version of [9]).

Second, for each measure μ in this class we do not require the algorithm to succeed on *all* sequences X that are μ-Martin-Löf random: we allow it to succeed only on a set of μ-probability $> \delta$, for a fixed positive δ independent of the measure μ.

Finally, the notion of success on a sequence X is now weaker: we do not require that the algorithm produces (for all sufficiently long inputs) some specific measure. Instead, we only ask that it gives 'good explanations' for the observed sequence from some point on. More specifically, we say that an algorithm $\mathfrak{A}$ *BD-succeeds* (BD stands for 'bounded deficiency') on some X, if for some c and for all sufficiently large n the measure $\mathfrak{A}(X\restriction n)$ is defined and X is random with deficiency at most c with respect to $\mathfrak{A}(X\restriction n)$. Clearly BC-success implies BD-success. (Recall that in our definition the randomness deficiency depends only on the measure but not on the algorithm that computes it.)

We now are ready to state the main result of this section, in its strong form.

Theorem 3.2. *Let $\mathcal{M}$ be a subset of $\mathcal{P}$ with the following properties*:

- *$\mathcal{M}$ is effectively closed, i.e., one can enumerate a sequence of rational open balls in $\mathcal{P}$ whose union is the complement of $\mathcal{M}$.*
- *$\mathcal{M}$ is recursively enumerable, i.e., one can enumerate all rational open balls in $\mathcal{P}$ that intersect $\mathcal{M}$.*
- *for every computable measure ν, and every non-empty open subset of $\mathcal{M}$ (i.e., a non-empty intersection of an open set in $\mathcal{P}$ with $\mathcal{M}$) there is a computable μ in this open subset which is orthogonal to ν.*

Let also δ be a positive rational number. Then there is no algorithm $\mathfrak{A}$ such that for every computable $\mu \in \mathcal{M}$, the μ-measure of sequences X on which $\mathfrak{A}$ BD-succeeds is at least δ.

The notion of a recursively enumerable closed set is standard in computable analysis, see [10, Definition 5.1.1].

Note that the hypotheses on the class $\mathcal{M}$ are not very restrictive: many standard classes of probability measures have these properties. In particular, the class $\{\beta_p \colon p \in [0,1]\}$ of Bernoulli measures is such a class; so there is no algorithm that can learn all Bernoulli measures (not to speak about all Markov chains). To see that the third condition is true, note that only countably many Bernoulli measures may be non-orthogonal to a given measure μ: the sets L_p of sequences with limit frequency p are disjoint, so only countably many of them may have positive μ-measure. It remains to note that every open non-empty subset of the class of Bernoulli measures has cardinality continuum.

Let us give another example beyond Bernoulli measures and Markov chains, where the probability of the next bit to be 0 may depend on many of the previous bits: for every parameter $p \in [0,1]$, consider measure μ_p associated to the stochastic process which generates a binary sequence bit by bit as follows: the first bit is 1, and the conditional probability of 1 after $\sigma 10^k$ is $p/(k+1)$. One can check that the class $\mathcal{P} = \{\mu_p : p \in [0,1]\}$ satisfies the hypotheses of the theorem (observe that p can easily be reconstructed from the sequence).

Note also that these hypotheses are not added for convenience: although they might not be optimal, they cannot be outright removed. If we do not require compactness, then the class of Bernoulli measures β_p with *rational* parameter p would qualify, but it is easy to see that this class admits an algorithm which correctly identifies each of the measures in the class with probability 1. The third condition is important, too. Consider the measures β_0 and β_1 concentrated on the sequences $0000\ldots$ and $1111\ldots$ respectively. Then the class $\mathcal{M} = \{p\beta_0 + (1-p)\beta_1 \ : \ p \in [0,1]\}$ is indeed effectively compact, but it is obvious that there is an algorithm that succeeds with probability 1 for all measures of that class (in the strongest sense: the first bit determines the entire sequence). For the second condition we do not have a counterexample showing that it is really needed, but it is true for all the natural classes (and it is guaranteed to be true if $\mathcal{M}$ has a computable dense sequence).

The rest of this section is devoted to the proof of Theorem 3.2.

Fix a subset $\mathcal{M}$ of $\mathcal{P}$ satisfying the hypotheses of the theorem, and some $\delta > 0$. We assume for the sake of contradiction that there is an algorithm

$\mathfrak{A}$ such that for every computable $\mu \in \mathcal{M}$, the μ-measure of sequences X on which $\mathfrak{A}$ BD-succeeds is at least δ. In the sequel, by "success" we always mean BD-success.

For every pair of integers (N, d), we define the set

$$\textsc{Wrong}(N,d) = \big\{X \big| (\exists n \geq N)\ \mathbf{d}(X|\mathfrak{A}(X\restriction n)) > d\big\}$$

This is the set of X's on which the algorithm is 'visibly wrong' at some prefix of length $n \geq N$, for the deficiency level d. Recall that $\mathfrak{A}(X\restriction n)$ may not converge to a measure, being a finite sequence of balls. In this case, as we have agreed, the inequality $\mathbf{d}(X|\mathfrak{A}(X\restriction n)) > d$ means that the $\mathbf{d}(X|\cdot)$ exceeds d everywhere on some closed ball provided by $\mathfrak{A}(X\restriction n)$ as an approximation for μ.

Note that $\textsc{Wrong}(N,d)$, understood in this way, is effectively open uniformly in (N,d) and is non-increasing in each of its parameters. The intersection of sets $\textsc{Wrong}(N,d)$ for all N and d is some set $\textsc{Wrong}$; as the name says, the algorithm $\mathfrak{A}$ cannot BD-succeed on any sequence in this set.

It is technically convenient to combine the two parameters N and d into one (even they are of different nature) and consider a decreasing sequence of sets $\textsc{Wrong}(N) = \textsc{Wrong}(N,N)$ whose intersection is $\textsc{Wrong}$.

We also consider a set $\textsc{Succ}(N,d)$ of all sequences X such that $\mathfrak{A}$ BD-succeeds on X at level N with deficiency d, i.e.,

$$\textsc{Succ}(N,d) = \{X\colon (\forall n \geq N)\ [\mathfrak{A}(X\restriction n) \text{ is a measure, } \mathbf{d}(X|\mathfrak{A}(X\restriction n)) \leq d]\}$$

The set $\textsc{Succ}(N,d)$ is a closed set. Indeed, it is an intersection of sets indexed by n, so we need to show that each of them is closed. For each n there are finitely many possible prefixes of length n, so the first condition ("$\mathfrak{A}(X\restriction n)$ is a measure") defines a clopen set. The second condition defines an effectively closed subset in each cylinder where $\mathfrak{A}(X\restriction n)$ is a measure. (Note that we do *not* claim that $\textsc{Succ}(N,d)$ is *effectively* closed, since the condition "to be a measure" is only a Π_2-condition.) By definition, the set $\textsc{Succ}(N,d)$ does not intersect the set $\textsc{Wrong}(N,d)$.

The set $\textsc{Succ}(N,d)$ increases as N or d increase; the union of these sets is the set of all X where $\mathfrak{A}$ succeeds; we denote it by $\textsc{Succ}$. Again we may combine the parameters and consider an increasing sequence of sets $\textsc{Succ}(N) = \textsc{Succ}(N,N)$ whose union in $\textsc{Succ}$.

All these considerations deal with the space of sequences. Now we switch to the space of measures and the class $\mathcal{M}$. We look what are the measures

of sets $\textsc{Wrong}(N)$ according to different $\mu \in \mathcal{M}$. Consider some threshold $x \in [0,1]$. There are two possible cases:

- there exist some number N, and some non-empty open set in $\mathcal{M}$ such that $\mu(\textsc{Wrong}(N)) \le x$ for all μ in this set.
- for every N the set of points $\mu \in \mathcal{M}$ where $\mu(\textsc{Wrong}(N)) > x$ is dense in $\mathcal{M}$.

There is some threshold where we switch from one case to the other, so let use take close values of $p < q$ (i.e., we take the difference $q - p$ to be much smaller than δ from the statement of the theorem; it would be enough to require that $q - p < \delta/10$) such that the first case happens for q and the second one happens for p.

Choose N and open ball $\mathcal{B}_0$ that has a non-empty intersection with $\mathcal{M}$ such that $\mu(\textsc{Wrong}(N)) \le q$ for all $\mu \in \mathcal{B}_0 \cap \mathcal{M}$ (this is possible since the first case happens for q).

Lemma 3.3. *There exists a computable measure $\mu^* \in \mathcal{B}_0 \cap \mathcal{M}$ such that $\mu^*(\textsc{Wrong}) \ge p$.*

Proof. Since the second case happens for p, we can find some $\mu \in \mathcal{B}_0 \cap \mathcal{M}$ such that $\mu(\textsc{Wrong}(0)) > p$. Since $\textsc{Wrong}(0)$ is open, the same is true for some its clopen subset C, i.e., $\mu(C) > p$. Note that $\mu(C)$ is a continuous function of μ for fixed clopen C, so we can find a smaller ball $\mathcal{B}_1 \subseteq \mathcal{B}_0$ intersecting $\mathcal{M}$ such the $\mu(\textsc{Wrong}(0)) \ge \mu(C) > p$ for all $\mu \in \overline{\mathcal{B}}_1 \cap \mathcal{M}$. Then, repeating the same argument, we find an even smaller ball $\mathcal{B}_2 \subseteq \mathcal{B}_1$ intersecting $\mathcal{M}$ such that $\mu(\textsc{Wrong}(1)) > p$ for all $\mu \in \overline{\mathcal{B}}_2 \cap \mathcal{M}$, then some $\mathcal{B}_3 \subseteq \mathcal{B}_2$ such that $\mu(\textsc{Wrong}(2)) > p$ for all $\mu \in \overline{\mathcal{B}}_3 \cap \mathcal{M}$, etc. Using the completeness of the space of measures, consider the intersection point μ^* of all $\mathcal{B}_i$ (we may assume that their radii converge to 0 and that $\overline{\mathcal{B}}_{i+1} \subseteq \mathcal{B}_i$, and this guarantees the existence and the uniqueness of the intersection point). We have $\mu^*(\textsc{Wrong}(i)) > p$ for all i (but $\mu^*(\textsc{Wrong}(N)) \le q$; the same in true for all subsequent sets $\textsc{Wrong}(i)$ for all $i \ge N$). The continuity property for measure μ^* then guarantees that $\mu^*(\textsc{Wrong}) \ge p$.

Refining this argument, we can get a *computable* measure μ^* with this property. Indeed, we may choose $\mathcal{B}_{i+1}$ in such a way that even the closed ball $\overline{\mathcal{B}}_{i+1}$ of the same radius is contained in $\mathcal{B}_i$; this property is enumerable. "To have a non-empty intersection with $\mathcal{M}$" is also an enumerable property (by

assumption), and "$\mu(\textsc{Wrong}(i)) > p$ for all $\mu \in \overline{\mathcal{B}}_{i+1}$" is also an enumerable property (we may assume without loss of generality that p is rational, and $\textsc{Wrong}(i)$ is effectively open uniformly in i). So we can perform a search until $\mathcal{B}_{i+1}$ is found, and the sequence of $\mathcal{B}_i$ is computable, so μ^* is computable. Lemma is proven.

Now the argument goes as follows. Since μ^* is computable, the set $\textsc{Succ}$ should have μ^*-probability at least δ by assumption. Success means that (at least) some measures are provided by the learning algorithm $\mathfrak{A}$ for prefixes of sufficiently large length M. There are finitely many possible prefixes, and they correspond to finitely many computable measures $\mu_1, \ldots, \mu_s$. Then we choose a measure μ' orthogonal to all these measures and very close to μ^* (this is possible according to our assumption). We get the contradiction showing that $\mu'(\textsc{Wrong}(N))$ is almost $p+\delta$ (or more) and therefore exceeds q which is not possible due to the choice of $\mathcal{B}_0$. To get the δ-increase we use the fact that sequences that are μ'-random cannot be μ_i-random and should therefore have infinite deficiency. Let us now explain this argument in details.

Recall that we have chosen N in such a way that $\mu(\textsc{Wrong}(N)) \le q$ for all μ sufficiently close to μ^*. On the other hand, $\mu^*(\textsc{Wrong}(M)) \ge \mu^*(\textsc{Wrong}) \ge p$ for all M.

Since $\mu^*(\textsc{Succ}) \ge \delta$, the continuity property of measures guarantees that $\mu^*(\textsc{Succ}(M)) \gtrsim \delta$ for sufficiently large M, where $\gtrsim$ means inequality up to an additive error term that is very small compared to δ (in fact, $\delta/10$ would be small enough; we do not add more than 10 inequalities of this type). Fix some M that is large enough and greater than N from the previous paragraph.

The set $\textsc{Wrong}(M)$ is open and has μ^*-measure at least p. Therefore, there exist a clopen set $C \subseteq \textsc{Wrong}(M)$ such that $\mu^*(C) \gtrsim p$. Since the set C is clopen, there exists some $K \ge M$ such that the K-bit prefix determines whether a sequence belongs to C (the granularity of C is at most K).

Now the Cantor space is split into 2^K intervals that correspond to different prefixes of length K. Some of these intervals form the set C (and belong to $\textsc{Wrong}(M)$ entirely). Among the rest, we distinguish good and bad intervals; good intervals correspond to prefixes for which the learning algorithm $\mathfrak{A}$ produces a measure (whatever this measure is). Let $\mu_1, \ldots, \mu_s$ be all measures that are produced by $\mathfrak{A}$ for all good intervals (we have at most 2^K of them).

Note that $\textsc{Succ}(M)$ is covered by the good intervals. Indeed, it is disjoint with $\textsc{Wrong}(M)$ and therefore with C, and also is disjoint with bad intervals by definition (since $K \geq M$, the algorithm $\mathfrak{A}$ should produce a measure when applied to K-bit prefix).

Now consider a measure μ' that is very close to μ and orthogonal to all μ_i. (Our assumption allows us to get a measure very close to μ and orthogonal to a given computable measure; now we have several measure $\mu_1, \ldots, \mu_s$ instead of one, but this does not matter since we may consider their average: any measure orthogonal to the average is orthogonal to all μ_i.)

Since the μ^*-measure of $\textsc{Succ}(M)$ is almost δ (or more), and it is covered by good intervals, then μ^*-measure of the union of good intervals is also almost δ (or more). The same is true for every measure μ' sufficiently close to μ^* since the union of good intervals is a clopen set.

No μ'-random sequences can be μ_i-random since the measures are orthogonal. This implies infinite deficiency, so all μ'-random sequences in good intervals belong to $\textsc{Wrong}(M)$. So the μ'-measure of the part of $\textsc{Wrong}(M)$ outside C is almost δ (or more), and the part of $\textsc{Wrong}(M)$ inside C has μ'-measure almost p or more (this was true for μ, and μ' is close to μ). Together we get lower bound close to $p + \delta$ for $\mu'(\textsc{Wrong}(M))$. And this gives us a contradiction, since $\mu'(\textsc{Wrong}(M)) \leq \mu'(\textsc{Wrong}(N))$, and the latter should be at most q for all μ' close to μ. (Recall that we have chosen $q - p$ much smaller than δ.)

This contradiction finishes the proof.

4. Removing the deficiency boundedness assumption: a positive result

Now we show that the bounded deficiency assumption in Theorem 3.2 cannot be omitted.

Theorem 4.1. *There exists an algorithm $\mathfrak{A}$ such that for every X which is Martin-Löf random with respect to some computable probability measure, and for almost all n, the value $\mathfrak{A}(X\restriction n)$ is a measure with respect to which X is Martin-Löf random.*

The proof of this result is inspired by a result of Harrington (reported in [4, Theorem 3.10]) which states that there exists a algorithm to learn all computable sequences *up to finitely many errors*. More precisely, there is an

algorithm $\mathfrak{A}$ such that for every computable sequence X, for almost all n, $\mathfrak{A}(X\restriction n)$ is defined (i.e., is an infinite sequence), and is equal to X except for finitely many bits. Indeed, consider a program $\mathfrak{A}(\sigma)$ that, given input m, spends time m searching for the minimal program computing some extension of σ and then runs this program, if found, on m (and returns, say, 0 if no such program is found).

PROOF. We will use an argument similar to Harrington's one to prove Theorem 4.1. In this section, it is more convenient to view measures as functions by identifying μ with the function $\sigma \mapsto \mu(\sigma)$ (here and in the rest of the section, $\mu(\sigma)$ is the abbreviation of $\mu([\sigma])$).

It is also more convenient in this section to use an alternative characterization of Martin-Löf randomness, via the Levin–Schnorr theorem, which states that if μ is a computable measure, a sequence X is Martin-Löf random with respect to μ if and only if the prefix complexity of its prefixes is big:

$$(\exists c)(\forall n)\ \mathrm{K}(X\restriction n) > -\log \mu(X\restriction n) - c$$

(see for example [7]). We say that measure μ is *exactly computable* when the function $\sigma \mapsto \mu(\sigma)$ is rational-valued and computable as a function from $2^{<\omega}$ to $\mathbb{Q}$. Of course, not all computable measures are exactly computable, but the following fact holds:

Lemma 4.2. *If $X \in 2^\omega$ is random with respect to a computable probability measure μ, it is random with respect to an exactly computable probability measure ν. Moreover, one can suppose that $\nu(\sigma) > 0$ for all strings σ.*

See [6] for a proof (essentially we approximate the given computable measure with enough precision using positive rational numbers).

This lemma is convenient because it is possible to effectively list the family $\mathcal{F}$ of partial computable functions μ from $2^{<\omega}$ to $\mathbb{Q}$ such that

- $\mu(\Lambda) = 1$;
- for every n, either $\mu(\sigma)$ is defined for all strings σ of length n, or is undefined for all strings σ of length n;
- if $\mu(\sigma 0)$ and $\mu(\sigma 1)$ are defined, $\mu(\sigma)$ is defined and is equal to $\mu(\sigma 0) + \mu(\sigma 1)$;

- $\mu(\sigma) > 0$ for all σ on which μ is defined.

Let $(\mu_e)_{e\in\mathbb{N}}$ be an effective enumeration of all the functions in $\mathcal{F}$. It is among these functions that, given a sequence X, we are going to look for the 'best candidate' μ_e such that μ_e is a measure (i.e., is total) and X is random relative to μ_e. Suppose we are given a prefix σ of X. What is a good candidate μ_e for this σ? For this, we use the same approach as algorithmic statistics: a good explanation μ_e for a string σ should (a) be defined on σ, (b) be simple, which is measured by the quantity $\mathrm{K}(e)$, and (c) give σ a small 'local' randomness deficiency, which we can measure by the quantity $\mathbf{ld}(e,\sigma) = \max_{\tau\preceq\sigma}[-\log\mu_e(\tau) - \mathrm{K}(\tau)]$. The value

$$\mathbf{d}(X\,|\,\mu_e) = \sup_{\tau\preceq X}[-\log\mu_e(\tau) - \mathrm{K}(\tau)] = \lim_n \mathbf{ld}(X\upharpoonright n\,|\,\mu_e) = \sup_n \mathbf{ld}(X\upharpoonright n\,|\,\mu_e)$$

is a version of randomness deficiency and is finite if and only if X is Martin-Löf random with respect to μ_e (see, e.g., [1] for details). Note that it is different from the uniform deficiency considered in the previous section, but since in this section we use only this notion, we use the same letter $\mathbf{d}$.

Returning to algorithmic statistic, we combine the two quantities into a score function

$$\mathbf{score}(e,\sigma) = \mathrm{K}(e) + \lceil \mathbf{ld}(e,\sigma)\rceil,$$

with $\mathbf{score}(e,\sigma) = \infty$ if $\mu_e(\sigma)$ is undefined (like in golf, 'score' is meant in a negative sense: high $\mathbf{score}(e,\sigma)$ means that μ_e is not a good candidate for being a measure with respect to which σ looks random). Finally, we define the function $\textsc{Best}$ such that $\textsc{Best}(\sigma)$ is the e which minimizes $\mathbf{score}(\sigma,e)$ (if there are several such e's, we take $\textsc{Best}(\sigma)$ to be the smaller one). The first thing to observe is that $\textsc{Best}$ is computable in the halting set $\mathbf{0}'$. Indeed, with access to $\mathbf{0}'$, one can first find an e such that $s = \mathbf{score}(\sigma,e) < \infty$ (because K is computable in $\mathbf{0}'$). Then, again using $\mathbf{0}'$, one can find an N such that $\mathrm{K}(e) > s$ (and thus $\mathbf{score}(e,\sigma) > s$) for all $e > N$, and then take $\textsc{Best}(\sigma)$ to be the e in $[0,N]$ which minimizes $\mathbf{score}(\sigma,e)$ (again taking the smallest one if there are several).

The core of the proof of Theorem 4.1 is the following lemma, which is of independent interest.

Lemma 4.3. *Let X be a sequence which is random with respect to some computable probability measure. The sequence of integers $\textsc{Best}(X\upharpoonright n)$ converges to a single value e^* such that μ_{e^*} is a measure, and X is random with respect to μ_{e^*}.*

Thus learning measures in the EX sense, which we showed in the previous section to be impossible, becomes possible if one is given access to oracle $\mathbf{0}'$.

To prove this lemma, consider the best explanation for the entire sequence, i.e., some integer e^* that minimizes the quantity

$$d = \mathrm{K}(e) + \lceil \mathbf{d}(X|\mu_e) \rceil$$

among all e such that μ_e is a measure. When looking for the best explanation for some prefix $X\restriction n$, we consider e too, but it may lose the competition to some other candidates, since we take into account only the local deficiency (and not its limit/supremum) and since these other candidates may have smaller complexity and deficiency on $X\restriction n$ but do not define a total measure. However, there are only finitely many other candidates that may be better than e^* since $\mathrm{K}(e)$ becomes too large for large e. And for sufficiently large values of n the local deficiency will reach its ultimate value, and all the non-total candidates are also disqualified, so e^* becomes the winner. Lemma 4.3 is proven.

At this point, we have seen that the function BEST does achieve the learning of measures we want, but unfortunately this function is only $\mathbf{0}'$-computable. By Schoenfield's limit lemma, this means that there exists a computable procedure which given σ generates a sequence $e_0, e_1, \ldots$ of integers which converges to $e_\infty = \textsc{Best}(\sigma)$. There is in general no way to compute μ_{e_∞} from this sequence. However, what we *can* do is assemble all the μ_{e_i} together (being cautious about the fact that some μ_{e_i}'s may be partial) into a single measure ν such that $\nu > c\mu_{e_\infty}$, and this, by definition of $\mathbf{d}$, guarantees that everything that is random with respect to μ_{e_∞}, is also ν-random.

More precisely, we have the following lemma.

Lemma 4.4. *Let $f : 2^{<\omega} \to \mathbb{N}$ be a total $\mathbf{0}'$-computable function such that $\mu_{f(\sigma)}$ is a measure for all σ. Then there is a computable function g such that $\mu_{g(\sigma)}$ is a measure for all σ, and $\mu_{g(\sigma)} \geq c_\sigma \mu_{f(\sigma)}$ for some positive c_σ.*

To do this, consider the following effective procedure. On input σ, we use Schoenfield's limit lemma to effectively get a sequence e_i converging to $e_\infty = f(\sigma)$. Initially all e_i are considered 'candidates'. We then apply a filtering process that deletes some of these candidates: We compute in parallel all $\mu_{e_i}(\tau)$ for all pairs (i, τ) for which e_i is still a candidate. If we find two candidates e_i, e_j and τ such that $\mu_{e_i}(\tau)$ and $\mu_{e_j}(\tau)$ are both defined and

different from each other, then we remove e_i and e_j from the list of candidates. This way we ensure, since the sequence converges, that from some point on, for any candidate e_i, the corresponding function μ_{e_i} is equal to μ_{e_∞} on its domain (but μ_{e_i} may be partial). Indeed, each bad candidate (i.e., an e_i such that $e_i \neq e_\infty$) may destroy at most one good candidate, and by assumption almost all candidates are good.

Now we let $\mu_{g(\sigma)}$ to be a computable measure ν constructed in the following way. First, let $\nu(\Lambda) = 1$. Then we compute the conditional probabilities $\nu(x0)/\nu(x)$ and $\nu(x1)/\nu(x)$ level by level. When computing them on level N, we use for the computation the conditional probabilities for some candidate that remains alive after N steps of filtering process. (Any of them could be used, for example, we may take the one with smallest computation time. Note the at least one good candidate remains, so we will not wait forever.)

As we have seen, starting from some level only good candidates remain, so the conditional probabilities above this level are the same for $\mu_{f(\sigma)}$ and ν. Since by assumption all values of all measures are positive, this guarantees the required inequality. This finishes the proof of Lemma 4.4.

We can now put all pieces together to prove Theorem 4.1. Applying the previous lemma to $f = \textsc{Best}$, we have a computable function g such that for every σ, the measure $\mu_{g(\sigma)}$ dominates, up to a multiplicative constant, the measure $\mu_{\textsc{Best}(\sigma)}$. For every X which is random with respect to some computable measure, we know, by Lemma 4.3, that $\mu_{\textsc{Best}(X\upharpoonright n)}$ is eventually constant and equal to a measure with respect to which X is random. This measure is dominated (up to multiplicative constant) by $\mu_{g(X\upharpoonright n)}$, thus X is also random with respect to $\mu_{g(X\upharpoonright n)}$ (change in the measure increases the deficiency at most by $O(1)$). This finishes the proof. □

Acknowledgements. Laurent Bienvenu and Santiago Figueira acknowledge the support of the Laboratoire International Associé "INFINIS". Laurent Bienvenu and Alexander Shen also acknowlegde the support of ANR-15-CE40-0016-01 RaCAF grant.

[1] Bienvenu, L., Gács, P., Hoyrup, M., Rojas, C., Shen, A., 2011. Algorithmic tests and randomness with respect to a class of measures. Proceedings of the Steklov Institute of Mathematics 274, 34–89.

[2] Bienvenu, L., Merkle, W., 2009. Constructive equivalence relations for computable probability measures. Annals of Pure and Applied Logic 160, 238–254.

[3] Bienvenu, L., Monin, B., Shen, A., 2014. Algorithmic identification of probabilities is hard. In: Algorithmic Learning Theory. Vol. 8776 of Lecture Notes in Computer Science. Springer, pp. 85–95.

[4] Case, J., Smith, C., 1983. Comparison of identification criteria for machine inductive inference. Theoretical Computer Science 25, 193–220.

[5] Gács, P., 2005. Uniform test of algorithmic randomness over a general space. Theoretical Computer Science 341 (1-3), 91–137.

[6] Juedes, D., Lutz, J., 1995. Weak completeness in E_1 and E_2. Theoretical Computer Science 143 (1), 149–158.

[7] Levin, L., 1984. Randomness conservation inequalities; information and independence in mathematical theories. Information and Control 61, 15–37.

[8] Li, M., Vitányi, P., 2008. An introduction to Kolmogorov complexity and its applications, 3rd Edition. Texts in Computer Science. Springer-Verlag, New York.

[9] Vitányi, P., Chater, N., 2013. Algorithmic identification of probabilities, http://arxiv.org/abs/1311.7385.

[10] Weihrauch, K., 2000. Computable analysis. Springer, Berlin.

[11] Zeugmann, T., Zilles, S., 2008. Learning recursive functions: a survey. Theoretical Computer Science 397, 4–56.