Distinguishing computable mixtures of quantum states

Ignacio H. López Grande,[1] Gabriel Senno,[2] Gonzalo de la Torre,[2] Miguel A. Larotonda,[1] Ariel Bendersky,[3,4]
Santiago Figueira,[3,4] and Antonio Acín[2,5]

[1]*DEILAP, CITEDEF-CONICET, Villa Martelli, Buenos Aires, Argentina*
[2]*ICFO-Institut de Ciencies Fotoniques, The Barcelona Institute of Science and Technology, 08860 Castelldefels (Barcelona), Spain*
[3]*Universidad de Buenos Aires, Facultad de Ciencias Exactas y Naturales, Departamento de Computación, 1428 Buenos Aires, Argentina*
[4]*CONICET-Universidad de Buenos Aires, Instituto de Investigación en Ciencias de la Computación (ICC), 1428 Buenos Aires, Argentina*
[5]*ICREA–Institucio Catalana de Recerca i Estudis Avançats, Lluis Companys 23, 08010 Barcelona, Spain*

(Received 10 August 2017; revised manuscript received 26 March 2018; published 7 May 2018)

In this article we extend results from our previous work [Bendersky *et al.*, Phys. Rev. Lett. **116**, 230402 (2016)] by providing a protocol to distinguish in finite time and with arbitrarily high success probability any algorithmic mixture of pure states from the maximally mixed state. Moreover, we include an experimental realization, using a modified quantum key distribution setup, where two different random sequences of pure states are prepared; these sequences are indistinguishable according to quantum mechanics, but they become distinguishable when randomness is replaced with pseudorandomness within the experimental preparation process.

DOI: 10.1103/PhysRevA.97.052306

I. INTRODUCTION

With the advance of the experimental realization of quantum protocols, the most widely used class of setups consists of classical systems controlling quantum ones [1–4]. Being classical, the control systems are limited in the type of operations they can perform, and this has implications on what can be achieved by the setups they control. In particular, as it was shown in [5], if one intends to prepare a maximally mixed state by means of a computer pseudorandomly choosing pure states from a given basis, there is an algorithm that can distinguish such a preparation from an adequately prepared maximally mixed state, without any knowledge of the mixing procedure.

In this work we extend the ideas from [5] in two ways. First, we generalize the theoretical result by showing that any preparation performed by a computer intended to generate the maximally mixed state, and not just those in which the states are chosen from a predefined basis, can be distinguished from the maximally mixed state. Second, we present an experimental implementation in which we distinguish two computable preparations that if carried out with randomness would be indistinguishable.

This article is organized as follows. First, we introduce the tools from the theory of algorithmic randomness which we will need later on. Second, we review the distinguishing protocol from [5]. Third, we present its generalization to arbitrary computable preparations. Finally, we present results for an experiment implementing a widely used scenario in which a pseudorandom function is used to pick pure states from a given basis.

II. PRELIMINARIES

Central to the distinguishing protocols we will describe in the following sections is the idea of an *algorithmically random* sequence of symbols. Roughly, an infinite sequence of symbols from some finite alphabet Σ is random in an algorithmic sense, if it lacks any regularity detectable by effective means. Randomness tests, also called *Martin-Löf tests (ML tests)* [6], are defined to detect some specific regularity. This "detection" of nonrandom sequences must be computably approximable, with incrementing levels of accuracy or significance. A test is a collection of sets V_m of possible prefixes of sequences that do not look random. As we increase m, the identification of nonrandomness gets more and more fine-grained, leaving in the limit a null measure set of nonrandom sequences. The *Martin-Löf random (ML-random)* sequences are those not detectable by any possible ML test.

Formally, let Σ^* be the set of all finite strings with symbols from Σ. A Martin-Löf test is a sequence $(V_m)_{m\in\mathbb{N}}$ of sets $V_m \subseteq \Sigma^*$ with the two following properties:

(1) *Effectiveness.* There is a Turing machine that, given m and i, produces the ith string of V_m (notice that in general there are infinitely many strings in V_m). It is not possible to computably determine if a string *is not* in V_m, but we can computably enumerate all strings that are in.

(2) *Null class.* Let λ be the uniform measure on the space Σ^ω of infinite sequences with symbols from Σ and, for $A \subseteq \Sigma^*$, let $[A] \subseteq \Sigma^\omega$ denote the set of sequences with prefixes in A. Then, we require each ML test $(V_m)_{m\in\mathbb{N}}$ to satisfy $\lambda[V_m] \leqslant |\Sigma|^{-m}$.

We say that a sequence $Y \in \Sigma^\omega$ is *ML random* if no ML test $(V_m)_{m\in\mathbb{N}}$ can capture Y in *all* its levels of accuracy, that is if for no ML test $(V_m)_{m\in\mathbb{N}}$ we have $Y \in \bigcap_m [V_m]$. Informally, if $Y \in [V_m]$ then we reject the hypothesis that Y is random with significance level $|\Sigma|^{-m}$. Observe that, if $Y \in [V_m]$, then there exist n such that the first n symbols of Y, denoted $Y \upharpoonright n$, belong to V_m.

An important feature of the theory of Martin-Löf randomness is the existence of a universal ML test, i.e., a ML test $(U_m)_{m\in\mathbb{N}}$ such that any sequence $Y \in \Sigma^\omega$ is ML random iff $Y \notin \bigcap_m [U_m]$. Since $\lambda \bigcap_m [U_m] = 0$, this implies that the set of ML-random sequences has measure 1. In other words, the sequence of independent throws of a $|\Sigma|$-faced dice is ML random with probability 1. It is important to note, however, that the existence of a universal ML test does not give rise to a general computable procedure to decide, in finite time and given a sequence Y, whether Y is ML random or not. Intuitively, this follows from the fact that, in finite time, only finitely many symbols $b_1 \dots b_n$ can be read and, so, there are always both infinitely many ML-random and infinitely many non-ML-random sequences extending $b_1 \dots b_n$. On the other hand, the existence of a universal ML test does give rise to a procedure P such that, given sequence Y, if Y is not ML random, P halts and detects this fact after seeing a sufficiently long prefix of Y. One such procedure P simply consists on enumerating the strings in the sets U_m and claiming that Y is not ML random if $Y \restriction n \in U_m$ for some n and m (notice that, when Y is ML random, P will either not halt or halt and give a wrong answer).

Intuitively, we expect a random sequence $Y(0)Y(1)Y(2)\cdots \in \Sigma^\omega$ to satisfy the law of large numbers, i.e.,

$$\lim_n \frac{|\{i < n | Y(i) = b\}|}{n} = \frac{1}{|\Sigma|} \quad \text{for all } b \in \Sigma. \tag{1}$$

Furthermore, it is natural to expect that for random sequences there should be no algorithmic way of *selecting* some subsequence of them not satisfying (1) (say, for instance, a subsequence of all zeros in the binary case). This property, known as *Church stochasticity* [7], is satisfied by ML-random sequences (see, e.g., Ref. [8, Section 2.5.]) and we will use this fact in what follows.

III. DISTINGUISHING PSEUDOMIXTURES OF QUANTUM STATES

In [5] we considered a scenario with two players, Alice and Bob, in which, first, Alice fixes a qubit basis, either the σ_z basis or the σ_x basis, and then, upon Bob's successive requests, pseudorandomly picks an eigenstate from the chosen basis and sends it to him. We gave a protocol for Bob to distinguish the (initially unknown to him) preparation basis in finite time and with arbitrarily high success probability. This implies that it is incorrect to characterize Bob's lack of knowledge about the preparation basis with the maximally mixed state as one would do if Alice were using randomness.

The protocol followed by Bob has two steps. First, he alternatively measures the qubits being sent by Alice in the σ_x and σ_z basis. This generates two binary sequences: X and Z (see Fig. 1 for a schematic description). When he measures in the preparation basis, the corresponding sequence will be a subsequence (either the odd or the even positions) of the pseudorandom sequence being used by Alice; when he measures in the other basis, the resulting bits are, according to quantum mechanics, independent flips of a fair coin and, therefore, they give rise to a ML-random sequence with probability 1. In the second step of the protocol, Bob uses a universal ML test $(U_m)_{m\in\mathbb{N}}$ to distinguish between these two kinds of sequences and hence find out the preparation basis. Namely, given a desired probability of error ϵ, he computes $m := \min_k[2^{-k} \leqslant \epsilon]$ and starts enumerating all the strings in $U_m = \{s_1, s_2, \dots\}$ until he finds some k such that for $Y = Z$ or $Y = X$ it happens that

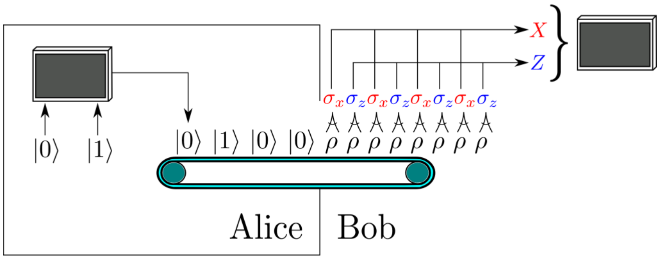
X
Z
$\sigma_x \sigma_z \sigma_x \sigma_z \sigma_x \sigma_z \sigma_x \sigma_z$
$\rho\ \rho\ \rho\ \rho\ \rho\ \rho\ \rho\ \rho$
$|0\rangle$ $|1\rangle$
$|0\rangle$ $|1\rangle$ $|0\rangle$ $|0\rangle$
Alice
Bob

FIG. 1. Schematic description of the protocol given in [5] allowing a player Bob to tell if he is being given pseudorandom eigenstates of the σ_x basis or of the σ_z basis.

$$Y \restriction n \in \{s_1, \dots, s_k\} \quad \text{for some } n \in \mathbb{N},$$

after which he claims that the box producing Y is the one with the computer. Since either X or Z is computable, and hence not ML random, the last condition has to be satisfied for sufficiently large n. His claim is wrong when the ML-random sequence was captured by $[U_m]$ before the computable one was (of course, for some $m' > m$ the ML-random sequence would be out of $[U_{m'}]$). Hence the probability of making this error is at most the probability for the coin flipping sequence to be inside $[U_m]$ and this is at most $2^{-m} \leqslant \epsilon$.

IV. GENERALIZED DISTINGUISHING PROTOCOL

In this section we extend the results from [5]. We will consider a scenario in which there are two boxes providing qudits to an observer named Bob. One of the boxes prepares single qudit maximally mixed states (for instance, by preparing the maximally entangled bipartite state $\frac{1}{\sqrt{d}} \sum_i |ii\rangle$ and keeping one half while providing the observer with the other). The other box contains an algorithm computing, on each round j, d complex algebraic numbers (i.e., roots of nonzero univariate polynomials with rational coefficients) [9] $\{c_k^{(j)}\}_{k\leqslant d}$ with $\sum_k |c_k^{(j)}|^2 = 1$ and preparing a qudit in the state $|\psi_j\rangle := \sum_k c_k^{(j)} |i\rangle$. Bob, without any knowledge about which box is which, will face the problem of determining the one preparing the maximally mixed state. Our main result is a protocol for Bob to win this game with arbitrarily high probability and independent of the program being run by the computer.

Before going to Bob's protocol, let us first note that if we fix a basis $\mathcal{B}$ and only allow the computer to pick eigenstates from such basis, a slight modification of the protocol from [5] allows Bob to distinguish between the boxes. Namely, if instead of alternating between measuring σ_x and measuring σ_z as in [5], Bob measures the outputs of both boxes in the $\mathcal{B}$ basis, the d-ary sequence associated with the box which has the computer will be computable and the other, according to quantum mechanics, independent tosses of a fair coin and so Martin-Löf random. Hence our previous result applies. The situation we want to consider in this work is when there is no fixed preparation basis.

Bob's protocol works as follows. In each round, he will perform an *informationally complete* positive operator valued measure (POVM) $\{E_i\}_{i\leqslant N_d}$ to the qudits coming out of each of the boxes satisfying

$$\mathrm{Tr}\left(E_i\frac{\mathbb{I}}{d}\right)=\frac{1}{N_d}\quad \text{for all } E_i. \tag{2}$$

It is easy to see that such POVMs with algebraic coefficients exist in every dimension d (for completeness, we provide a proof of this fact in Appendix A). This will give rise to two N_d-ary sequences B_1 and B_2 formed by the results of the measurements over the qudits coming from boxes 1 and 2. Note at this point that although Bob measures finitely many times, the sequences are infinite in the sense that he can keep requesting qudits from both boxes and making as many measurements as he needs. As we will see now, sequences B_1 and B_2 have a distinctive feature that will allow Bob to distinguish which is the maximally mixed state and which is the one being produced by a computer.

Let $r \in \{1,2\}$ be the box preparing the maximally mixed state and $c = 3 - r$ be the box with the computer inside. It follows from (2) that when measuring the POVM over the maximally mixed state $\frac{\mathbb{I}}{d}$, as all the effects are equiprobable, the resulting sequence B_r will consist of independent samples from the uniform distribution over $\{1,\dots,N_d\}$ and hence will be Martin-Löf random with probability 1. On the other hand, with probability 1, sequence B_c will not be Martin-Löf random. This is not straightforward, and we prove it next.

First, from the fact that the POVM $\{E_i\}_{i\leqslant N_d}$ satisfies (2) and it is informationally complete, notice the following.

Observation 1. Let $|\psi_j\rangle$ be the pure state produced by box c at round j. There is, at least, one E_i such that $\mathrm{Tr}(E_i|\psi_j\rangle\langle\psi_j|) > 1/N_d$.

This, together with the following lemma, will allow us to show that any computable preparation made by Alice is distinguishable from the correctly prepared maximally mixed state.

Lemma 1. With probability 1, sequence B_c is not ML random.

Proof. Following Observation 1, without loss of generality, we assume that E_k is such that

$$\mathrm{Tr}(E_k|\psi_n\rangle\langle\psi_n|) > 1/N_d \quad \text{for infinitely many } n. \tag{3}$$

This means that there is an algorithmic way to identify a subsequence of B_c not satisfying the law of large numbers (with probability 1). Namely, let $h:\mathbb{N}\to\mathbb{N}$ be defined as

$$h(0) := 0,$$

$$h(n+1) \coloneqq \min_m\left[\left(\mathrm{Tr}(E_k|\psi_m\rangle\langle\psi_m|) > \frac{1}{N_d}\right)\wedge[m > h(n)]\right].$$

By assumption (3), $h(n)$ is defined for all n. Next, by definition of h, with probability 1 the sequence

$$Y = B_c(h(0))B_c(h(1))B_c(h(2))\cdots \in \{1,\dots,N_d\}^{\omega},$$

which is a subsequence of B_c, does not satisfy the law of large numbers (1). Hence, noting that $|\psi_m\rangle$ is computable from m (e.g., with Alice's program) and so h is a computable function, we have that, with probability 1, B_c is not Church stochastic and so it is also not ML random.

We have proven that B_c is not ML random but B_r is. Now the argument carries on as in [5]. Namely, given a desired probability of error ϵ, Bob computes $m := \min_k[2^{-k} \leqslant \epsilon]$ and starts enumerating all the strings in $U_m = \{s_1,s_2,\dots\}$ until he finds some n such that $[B_i \restriction n] \subseteq \bigcup_{i\leqslant n}[s_i]$ for some $i \in \{1,2\}$ and claims that box i is the one with the computer. Since, with probability 1, either B_1 or B_2 is not Martin-Löf random, the last condition has to be satisfied for sufficiently large n with probability 1. His claim is incorrect when the sequence ML random was captured by $[U_m]$ which happens with probability $2^{-m} \leqslant \epsilon$.

V. EXPERIMENTAL TEST

In this section we present a proof-of-concept realization of the distinguishing protocol presented in [5] and resumed above. In the next lines we describe the additions or modifications made to the theoretical scenario, arising from experimental considerations.

First, to account for experimental imperfections, we will work under the assumption of a noise model consisting of a flip probability f in the observed symbols. That is, we consider the situation in which those results obtained when measuring the qubit states in the actual basis used by Alice are correct with probability $1 - f$ (this simple noise has no effect on the results of measurements performed in the wrong basis). This is a natural noise model in which random bit flips are applied to the measured sequences, resulting for instance from imperfect preparations or measurements. Noisy channels like the depolarizing channel and the bit-phase-flip channel can produce such an effect on the set of states used for this protocol.

For the sake of concreteness, we describe next an explicit algorithm for Bob to distinguish which of the sequences of measurement outputs X and Z is the one corresponding to measuring in the preparation basis (see Fig. 1). This algorithm, although less resistant to noise than the general protocol using ML tests given in [5], is robust enough for the noise model we are considering.

Bob will dovetail between program number and the maximum time steps required for the simulation of this program on a (fixed) universal Turing machine $\mathbf{V}$ (that is, he will simulate program 1 for one time step, then programs 1 and 2 for two time steps, and so on). This is a common technique in computability theory to avoid nonhalting programs (see, e.g., Ref. [10]). For each program p of length $|p|$ he will compute the Hamming distance (i.e., the number of different bits) between its output at time t and the first $k|p|$ bits of the sequences X and Z (notated $X \restriction k|p|$ and $Z \restriction k|p|$, respectively). The parameter $k \in \mathbb{N}$ will depend on the probability of success we are looking for. Whenever he finds a match for the first $k|p|$ bits, he halts and claims that the corresponding sequence is the computable one. Letting $q \in \mathbb{Q}$ be the fraction of bit flips in the prefixes, the pseudocode is Algorithm 1 below, where d_H denotes Hamming distance.

Algorithm 1 The noise tolerant distinguishing protocol

```
Input: $q \in \mathbb{Q}$, $k \in \mathbb{N}$ and $X,Z \in \{0,1\}^{\omega}$, one of them being
  computable
Output: "$X$" or "$Z$" as the candidate for being computable;
  wrong answer with probability bounded by $O(2^{-k})$
  for $t = 0,1,2,\dots$ do
    for $p = 0,\dots,t$ do
      If $d_H(\mathbf{V}_t(p), X \restriction k|p|) < qk|p|$ than
        output "$X$" and halt
      If $d_H(\mathbf{V}_t(p), Z \restriction k|p|) < qk|p|$ than
        output "$Z$" and halt
```

In Appendix B we show that the probability of error, i.e., the probability of Bob making a wrong claim about which of the two sequences X and Z is a subsequence (with its bits flipped with probability q) of Alice's sequence, is

$$P_{\text{err}} < \frac{2^{1+qk-k}\left(\frac{e}{q}\right)^{qk}}{1 - 2^{1+qk-k}\left(\frac{e}{q}\right)^{qk}} \tag{4}$$

and it can be shown numerically that for $q \lesssim 0.21$ it goes to zero exponentially with k. This distinguishing protocol appeared in a first version [5] of the results published in [5].

Notice that Algorithm 1—as it was the case with the protocol using a universal ML test—is independent of Alice's algorithm. This independence, however, comes at the expense of unfeasibility, because it is achieved through a search over the whole space of all Turing machines. Hence the second implementation decision we make is to restrict the possible algorithms used by Alice to the *rand*() function of Matlab using the Mersenne Twister default generator algorithm [11] with initial binary string seeds of a fixed maximum length $\ell_{\max}$ [technically, the seeds for the *rand*() function are nonnegative integers, but we identify them with binary strings in the usual encoding]. In spite of being a simplified scenario, this still represents a quite usual experimental situation. Finally, some minor changes to Algorithm 1 were required due to the nondeterministic nature of the emission and detection of *Poissonian* single photon states used as physical implementation for qubits. The adapted protocol can be specifically stated as follows:

(i) Alice and Bob set the value of two parameters from the protocol: $\ell_{\max}$ which determines the maximum length of the *rand*() function seed to be used and k which bounds to $N = k \times \ell_{\max}$, the number of qubits to be transmitted on any run of the experiment.

(ii) Alice pseudorandomly chooses an l-bit string s with $l \leqslant \ell_{\max}$ which is used as the initial value, or seed, for the *rand*() function. Then, she runs the *rand*() function N times, giving rise to N values $r_i \in [0,1]$ which she then rounds to the nearest integer (zero if $r_i \leqslant 1/2$ and 1 otherwise) and concatenates to form a string of N *pseudorandom* bits. Henceforth, we will denote an M-bit string constructed in this way using a string s as initial seed to the *rand*() function by $binrand(M,s)$.

(iii) Alice chooses randomly (with fair coin randomness as explained below) the basis in which she will encode and send the string.

(iv) Alice sends the N qubits to Bob. She encodes the binary string information in the photon polarization degree of freedom of a faint pulsed light beam.

(v) Bob measures the $\frac{N}{2}$ even and $\frac{N}{2}$ odd elements, each in one of the mutual unbiased bases.

(vi) Bob, after measurement, computes the Hamming distance (for even and odd bits) between the experimental data and $binrand(M,s)$ for different seeds s and increasing length M. When the minimum Hamming distance condition is fulfilled, Bob ends the search.

(vii) Finally Bob compares the state preparation (σ_x or σ_z mixtures) predicted by him with the mixture that was actually prepared by Alice to estimate the error probability (P_{err}) of the prediction.

A complete experiment consists in several repetitions of the protocol sketched above. Every execution is divided in two parts: the *transmission* of qubits from Alice to Bob, followed by a *search* routine, where Bob compares both bit strings with the pseudorandom strings generated with the *rand*() function over all seeds of length bounded by $\ell_{\max}$ as it is stated in the theoretical protocol. When Bob finds a string that resembles the experimental series up to a certain d_H value, the search ends. The result is compared with the actual basis used by Alice and the wrong guesses are registered as errors. After this they repeat the procedure with a new seed pseudorandomly picked and a new random emission basis choice. The bound for d_H allows us to control the tolerance of the experiment against the quantum bit error rate (QBER).

One thing to be noticed is that Bob may not find a series that fulfills the desired Hamming distance condition. This is a situation that is not present in the theoretical protocol. In this way every time that Bob doesn't find a match we compute the experiment as inconclusive and it is discarded. To overcome this issue, the parameters of the protocol (such as maximum d_H allowed) were set to guarantee that the probability of error occurrence was always greater than the probability of not finding any bit string fulfilling the condition. Under such assumptions, and using reasonable tolerances, we find that the ratio of inconclusive experiments to total number of errors was negligible.

A. Experimental setup

The above protocol was tested on a photonic setup, based on a modified BB84 quantum key distribution (QKD) implementation [12] which consists of an emission stage that is able to send binary states coded in two different unbiased bases of the photon polarization, which are called computational basis and diagonal basis, and a reception stage for the quantum channel. Additionally, a classical communication channel is added for synchronization, transmission, and data validation.

The four polarization qubits are obtained using attenuated coherent pulses generated with four infrared LEDs, controlled by a fast pulsed driver (optical pulse duration 25 ns FWHM). Faint coherent pulses can be used as probabilistic single photon sources: on each pulse the photon number distribution is Poissonian. Unlike the theoretical protocol, where each qubit is sent and received deterministically, here the transmission of a qubit is probabilistic. As opposed to QKD, in this demonstration the fact that most of the emitted pulses have zero photons requires Alice to send each state several times until Bob makes a successful detection.

Polarization states are obtained by combining all the outputs from the LEDs in a single optical path using polarization beam splitters (PBS), a half-wave plate retarder, and a beam splitter (BS). A bandpass filter centered at 810 nm narrows the photon's bandwidth down to 10 nm FWHM. A TTL clock pulse is sent to Bob every time a pulse is emitted in order to synchronize the optical pulses with the gated detection scheme.

At Bob's side the detection basis is passively and randomly selected with a BS. Each detection basis consists in a PBS with both outputs coupled into multimode fibers. An additional half-wave plate before one of the PBS allows for detection in the diagonal basis. We implement a polarization to time-bin transformation by adding different delays to each channel. This allows us to utilize a fiber multiplexing scheme with only one single photon detector (Fig. 2). Temporal masks generated

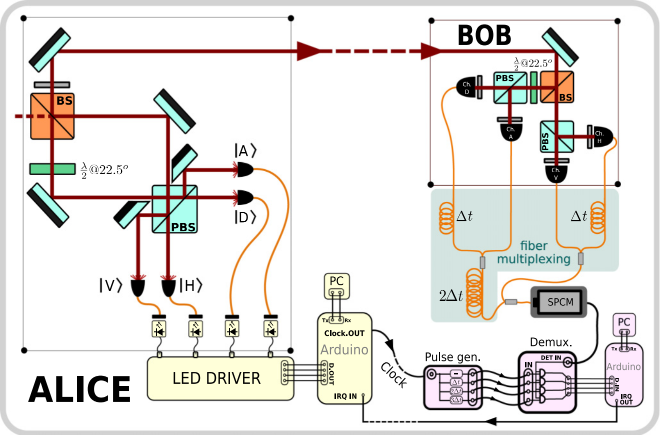

FIG. 2. Complete setup for implementing the transmission and search protocol: qubits encoded in polarized faint pulses are produced by infrared LEDs. Light is coupled into and decoupled from multimode fibers to obtain uniform beams for the four sources. The polarization state preparation is achieved by passing through a PBS (for H and V states) and an extra half-wave plate for the D and A paths. A nonpolarizing beam splitter cube couples the optical paths into an only exit light path. At the receiver's side a BS passively and randomly selects the detection basis for each incoming pulse. The outputs are coupled into multimode optical fibers, where different delays are imposed to make a polarization to time-bin transformation into a common output fiber. Finally a photon counter module and a temporal mask demultiplexer are used for detection.

using the clock pulse emitted by Alice act as demultiplexer and detection gating.

Programmable Arduino Mega 2560 boards are used to carry out the synchronization, communication, and data processing tasks, for which specific interfacing peripherals were developed. A desktop personal computer generates the binary strings of pseudorandom bits using the Matlab function *rand*() and stores the bit strings. Finally a quantum random number generator (QRNG) based on which-path detections of single photons exiting a beam splitter is used for the realization of a random selection of the emission basis on each repetition of the experiment.

Alice sends each bit of the string repeatedly at a frequency of 170 kHz until she receives an interruption signal, indicating that the qubit was correctly detected by Bob. Due to the probabilistic nature of the qubit transmission process each state may be sent several times before Bob makes a successful detection. In particular, given that the photon number distribution per pulse is Poissonian (with a mean photon value at the detector of 0.1), on average one every ten pulses is detected. Furthermore, the detection base is randomly selected so 50% of the detected photons are discarded by base mismatch. This results in an overall qubit transmission rate of $\frac{1}{20}$ per emitted pulse.

For this reason, every sent bit is registered by Bob (in the corresponding basis, either in the correct state or with an error), and as a side effect, we do not need to take losses into account in the noise model. This characteristic of the experimental protocol allows a simpler postprocessing of the data while not having any essential implication in the distinguishability between the resulting bit strings.

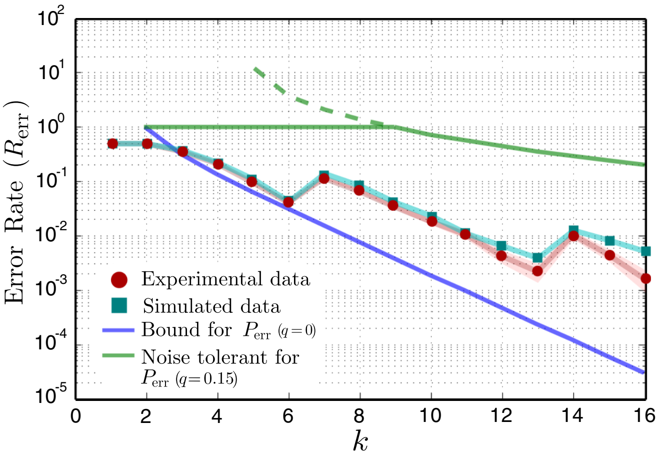

FIG. 3. Experimentally estimated error rate (the error that Bob does make a wrong claim) obtained with the noise tolerant protocol (circles over red lines), compared with the theoretical bounds for the noiseless (lower line in blue) and noise tolerant (upper line in green) algorithms. The cyan line with squares is the computational simulation of the experimental data taking into account the average QBER.

B. Complete results and simulations

Herein we analyze the experimental results. We compare the performance of Bob at guessing the emission basis, with the error probability $P_{\rm err}$ obtained in [5], and we also present additional data analysis aiming to explain the behavior of the error rate obtained.

The experiment involved 3100 repetitions of the *transmission* and *search* protocols. The total number of qubits transmitted on each repetition was fixed, and set by $k \times \ell_{\rm max}$ (in this implementation $\ell_{\rm max} = 10$). The parameter k determines the theoretical error probability for a given tolerance (q) and was set to take values between 1 and $k_{\rm max} := 16$. This bounds the maximum number of compared bits on each Hamming distance calculation to $N = 320$ ($\ell_{\rm max} \times k_{\rm max}$ bits for even and odd bits); that is the number of qubits that Alice sends to Bob on each run.

As a result of each run, Bob gets two 160-bit length strings M_e and M_o. M_e are the outcomes of even qubits, measured in the computational basis, and M_o are the outcomes of odd qubits, measured in the diagonal basis. These two strings correspond to Z and X introduced in Algorithm 1. Bob then searches for the first string s (in the lexicographical order) with $1 \leqslant |s| \leqslant \ell_{\rm max}$ such that

$$d_H(binrand(k|s|,s), M_i \upharpoonright k\,|s|) \leqslant \lfloor q \times k \times |s| \rfloor \quad (5)$$

for $i = o$ or $i = e$. In this experiment, the tolerance parameter is set to $q = 0.15$. If (5) is satisfied for $i = o$, he guesses that the preparation basis is the computational basis; else, if (5) is satisfied for $i = e$, he guesses that it is the diagonal basis. As explained before, if he doesn't find a seed s such that (5) is satisfied, the run is reported as inconclusive. The result of the search for each run is registered for a further estimation of the error rate $R_{\rm err}(k,q)$.

The probability of error in Bob's guess of the emission basis can be estimated for different values of the parameter k. Figure 3 shows the error rate obtained from the experimental

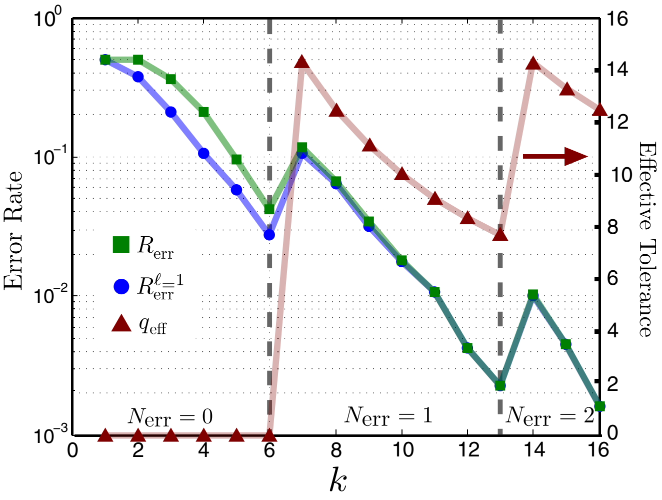

FIG. 4. Green curve (squares) shows the total wrong claim probability (error rate) and the contribution to this quantity coming from the programs of length 1 (blue dots), together with the effective tolerance $q_{\rm eff}$ (red triangles) for each value of k over the 3100 experiment repetitions. The tolerance value q was set to 0.15. Almost every error comes from $\ell = 1$ programs and the probability of error occurrences coming from programs with $\ell > 1$ vanishes as k increases. The sudden increases of $q_{\rm eff}$ in $k = 7$ and $k = 14$ appear due to the discrete nature of the maximum number of errors allowed on an accepted bit string (maximum Hamming distance). Also for each value of k where the effective tolerance probability for $\ell = 1$ increases, the error probability also increases. The vertical dashed lines delimit the regions where the maximum number of bit flips $N_{\rm err}$ allowed (for $\ell = 1$) is constant.

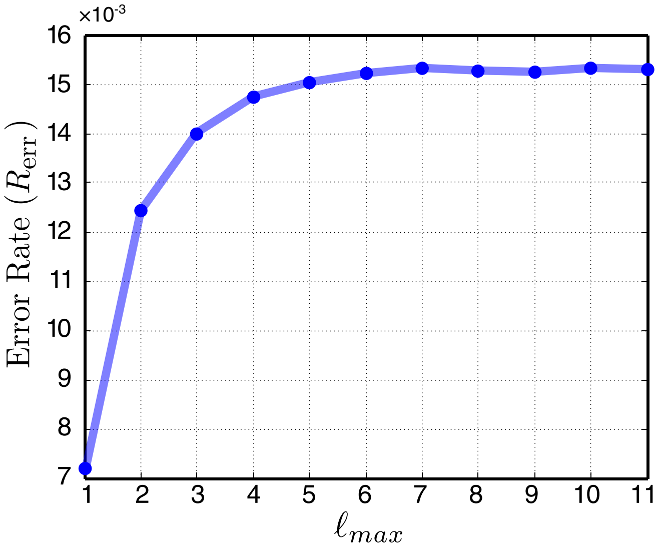

FIG. 5. Plotted data correspond to the wrong claim probability obtained by doing simulations of the experiment with an increasing set of programs (from 2^1 to 2^{11} programs). For each value of $\ell_{\rm max}$ we run a simulation of the experiment with 5×10^4 repetitions. The $P_{\rm err}$ value stabilizes as the number of programs grows.

data and from a computational simulation of the experiment, together with the theoretical bounds for the distinguishing—noiseless and noise tolerant—protocols.

The error rate as a function of k remains always below the noise-tolerant limit and also above the noiseless theoretical bound (excluding the scenarios with values $k = 1$ and $k = 2$). The error shows some unexpected increments for $k = 7$ and $k = 14$. This behavior arises due to the discrete nature of the number of errors allowed on each string comparison, and it is explained below.

Figure 4 shows the total error rate and the contribution to this quantity arising from programs of length 1. It is evident that errors occur mostly in the minimum length programs. In particular for $k > 10$ all the guessing errors come from these programs (which are the first to be evaluated in the *search* procedure). This fact simplifies the description of the error occurrence just in terms of the probability of error occurrence while evaluating length 1 programs.

The tolerance q determines the maximum number of errors allowed: $N_{\rm err} = \lfloor q \times k \times \ell \rfloor$. This quantity divided by the program length gives the *effective tolerance*: $q_{\rm eff} = \frac{\lfloor q \times k \times \ell \rfloor}{\ell}$. As almost all the errors arise from minimum length programs, the $R_{\rm err}$ increments can be explained looking at $q_{\rm eff}$ from $\ell = 1$. As can be seen in Fig. 4, for k below 6 the effective tolerance is zero ($N_{\rm err} = 0$). That is why the error rate follows the ideal theoretical curve for these values (Fig. 3). The increments on the error at $k = 7$ and $k = 14$ are correlated with increments in the $q_{\rm eff}$ (this will happen for every k where the number of maximum bit flips allowed $N_{\rm err}$ is increased by 1 for $\ell = 1$ programs).

Finally, as a validation of the results, we simulated the same experiment with different sizes of the set of programs used for fixed values of $k = 14$ and $q = 0.15$ (recall that for this experiment the program list was restricted to 2^{10} different elements). Figure 5 shows the simulated error rate for values of the maximum length program $\ell_{\rm max}$ from 1 to 11 over 5×10^4 repetitions. The error rate stabilizes as $\ell_{\rm max}$ grows. This shows that our results are representative of the values that would be obtained if an experiment with larger $\ell_{\rm max}$ was performed. In this regard, a similar experiment was implemented afterwards, utilizing a larger set of seeds for the *rand()* function: the complete protocol was implemented with $\ell_{\rm max} = 16$ (65536 different seeds) over 3200 repetitions of the experiment with $q = 0.15$ and k taking values from 1 to 16 showing the same behavior of the basis guess success rate.

VI. DISCUSSION

In this article we extended results from [5] in two ways. First, we proved that any attempt to mix pure states into the maximally mixed state, when performed by a computer (or any system equivalent in terms of computability power), can be distinguished from the maximally mixed state prepared correctly (either the one obtained by looking at a part of a maximally entangled state or by using a truly random source). This broadens the scope of the previous results, in which only some computable mixtures were analyzed. It should be noted that, as was the case with the protocol in [5], the distinguishing procedure presented here is, although computable, computationally expensive (a necessary price to pay for its generality). This implies that, in a cryptographic setting in which the attacker is usually restricted to polynomial time computations, the presented distinguishing procedure possesses no threat.

Second, we presented a proof-of-concept experiment showing that mixing two different sets of pure states that are supposed to yield the same mixed state can be distinguished when mixed employing one of the most widely used general purpose pseudorandom number generators.

These two results should be seen as a call for attention when performing experiments and claiming to produce certain mixed states via computable mixings. Furthermore, as the experimental proof-of-concept we provide shows, this persistence of the "algorithmic signature" in states comprising the computable mixings can be readily seen in standard QKD setups, which should also be taken into account as a weakness when using those setups for QKD.

ACKNOWLEDGMENTS

This work is supported by the Argentinian ANPCyT (Grants No. PICT-2014-3711 and No. PICT 2015-2293) and CONICET (Grant No. PIP 2014-0648), the Laboratoire International Associée INFINIS, the ERC CoG QITBOX, the AXA Chair in Quantum Information Science, the Spanish MINECO (QIBEQI Grant No. FIS2016-80773-P and Severo Ochoa Grant No. SEV-2015-0522), Generalitat de Catalunya (CERCA Programme and SGR 1381), and Fundació Privada Cellex. The authors would also like to thank Laura Knoll and Christian Schmiegelow for fruitful discussions.

APPENDIX A: POVM FOR THE GENERALIZED DISTINGUISHING PROTOCOL

For completeness, in this section we describe an informationally complete POVM $\{E_i\}_{i\leqslant N_d}$ satisfying (2). We construct it from the following $N_d := d(2d-1)$ projectors:

$$\begin{aligned}
\Pi^{(a)}_{m} &:= |m\rangle\langle m|,\\
\Pi^{(b\pm)}_{n,m} &:= \tfrac{1}{2}[|m\rangle\langle m| \pm |m\rangle\langle n| \pm |n\rangle\langle m| + |m\rangle\langle m|],\\
\Pi^{(c\pm)}_{n,m} &:= \tfrac{1}{2}[|m\rangle\langle m| \mp i|m\rangle\langle n| \pm i|n\rangle\langle m| + |m\rangle\langle m|],
\end{aligned}$$

for all $m < n \leqslant d$. It is easy to see that (1) $\mathrm{Tr}(\Pi^{(a)}_{m}\frac{\mathbb{I}}{d}) = \mathrm{Tr}(\Pi^{(b\pm)}_{n,m}\frac{\mathbb{I}}{d}) = \mathrm{Tr}(\Pi^{(c\pm)}_{n,m}\frac{\mathbb{I}}{d}) = \frac{1}{d}$ and (2) for every density matrix ρ over $\mathbb{C}^d$,

$$\begin{aligned}
\rho_{m,m} &= \mathrm{Tr}\left(\Pi^{(a)}_{m,n}\rho\right),\\
\rho_{m,n} &= \tfrac{1}{2}\big[\mathrm{Tr}\left(\Pi^{(b+)}_{m,n}\rho\right) - \mathrm{Tr}\left(\Pi^{(b-)}_{m,n}\rho\right)\\
&\quad + i(\mathrm{Tr}\left(\Pi^{(c-)}_{m,n}\rho\right) - \mathrm{Tr}\left(\Pi^{(c+)}_{m,n}\rho\right)\big] \quad \text{for } m \neq n.
\end{aligned}$$

Finally, since

$$\sum_{n,m}\left[\Pi^{(a)}_{n,m} + \Pi^{(b\pm)}_{n,m} + \Pi^{(c\pm)}_{n,m}\right] = (2d-1)\mathbb{I},$$

by normalizing these projectors with $1/(2d-1)$ we get the the effects E_i of a POVM with the desired characteristics.

APPENDIX B: PROBABILITY OF SUCCESS OF ALGORITHM 1

We need to bound the number of sequences that have a Hamming distance smaller than $qk\ell$ from a computable one. One possible bound is $2^{\ell}\binom{\ell k}{\lfloor q\ell k\rfloor}2^{\lfloor q\ell k\rfloor}$, where the first exponential term counts the number of different programs of length ℓ, the combinatorial number corresponds to the number of bits that can be flipped due to errors, and the last exponential term gives which of these bits are actually being flipped. This estimation may not be tight, as we may be counting the same sequence several times. However, using this estimation we derive a sensible upper bound for the final error probability, as we get

$$P_{\text{err}} < \sum_{\ell>0}\frac{2^{\ell}2^{\lfloor q\ell k\rfloor}\binom{\ell k}{\lfloor q\ell k\rfloor}}{2^{\ell k}}. \tag{B1}$$

If we consider that $q < 1/2$, we can remove the integer part function and use the generalization of combinatorial numbers for real values. Then, by using that $\binom{a}{b} \leqslant \left(\frac{ea}{b}\right)^{b}$, we obtain

$$P_{\text{err}} < \sum_{\ell>0}\left[2^{(1+qk-k)}\left(\frac{e}{q}\right)^{qk}\right]^{\ell}. \tag{B2}$$

This geometric sum can be easily computed yielding

$$P_{\text{err}} < \frac{2^{1+qk-k}\left(\frac{e}{q}\right)^{qk}}{1-2^{1+qk-k}\left(\frac{e}{q}\right)^{qk}}. \tag{B3}$$

Now it can be numerically shown that for $q \lesssim 0.21$ the probability of misrecognition tends to zero exponentially with k.

Finally, for completeness, we show that (with probability 1) Algorithm 1 halts for all inputs satisfying the assumptions. Let $f < q$ be the probability of a bit flip. With probability 1, we have that for every δ there exists an m_0 such that for every $m > m_0$ the portion of bit flips in both $X \upharpoonright m$ and $Z \upharpoonright m$ are less than $(f+\delta)m$. This means that if we go to long enough prefixes (or programs), the portion of bit flips will be less than q. And since any computable sequence is computable by arbitrarily large programs, this ensures that our algorithm will, at some point, come to an end.

[1] R. Prevedel, P. Walther, F. Tiefenbacher, P. Böhi, R. Kaltenbaek, T. Jennewein, and A. Zeilinger, Nature (London) **445**, 65 (2007).

[2] S. Takeda, T. Mizuta, M. Fuwa, P. van Loock, and A. Furusawa, Nature (London) **500**, 315 (2013).

[3] M. Barrett, J. Chiaverini, T. Schaetz, J. Britton, W. Itano, J. Jost, E. Knill, C. Langer, D. Leibfried, R. Ozeri *et al.*, Nature (London) **429**, 737 (2004).

[4] H. Takesue, E. Diamanti, T. Honjo, C. Langrock, M. Fejer, K. Inoue, and Y. Yamamoto, New J. Phys. **7**, 232 (2005).

[5] A. Bendersky, G. de la Torre, G. Senno, S. Figueira, and A. Acín, Phys. Rev. Lett. **116**, 230402 (2016).

[6] P. Martin-Löf, Inf. Control **9**, 602 (1966).

[7] A. Church, Bull. Am. Math. Soc. **46**, 130 (1940).

[8] L. Ming and P. M. Vitányi, *Kolmogorov Complexity and Its Applications* (Elsevier, Amsterdam, 2014), p. 187.

[9] Restricting to this subfield of the field of complex numbers stems from the fact that we need the field operations $(+,\cdot)$ as well as the standard order $<$ over the real subset to be computable. Of course, any other subfield of the complex numbers satisfying these requirements will do as well.

[10] M. Davis, R. Sigal, and E. J. Weyuker, *Computability, Complexity, and Languages: Fundamentals of Theoretical Computer Science* (Newnes, Oxford, 1994).

[11] M. Matsumoto and T. Nishimura, ACM Trans. Model. Comput. Simul. (TOMACS) **8**, 3 (1998).

[12] I. H. López Grande, C. T. Schmiegelow, and M. A. Larotonda, Pap. Phys. **8**, 080002 (2016).