Randomness Extraction from Bell Violation with Continuous Parametric Down-Conversion - Supplemental Material

Lijiong Shen,[1, 2] Jianwei Lee,[1] Le Phuc Thinh,[1] Jean-Daniel Bancal,[3] Alessandro Cerè,[1] Antia Lamas-Linares,[4, 1] Adriana Lita,[5] Thomas Gerrits,[5] Sae Woo Nam,[5] Valerio Scarani,[1, 2] and Christian Kurtsiefer[1, 2]

[1]*Centre for Quantum Technologies, National University of Singapore, 3 Science Drive 2, Singapore 117543*
[2]*Department of Physics, National University of Singapore, 2 Science Drive 3, Singapore 117551*
[3]*Department of Physics, University of Basel, Klingelbergstrasse 82, 4056 Basel, Switzerland*
[4]*Texas Advanced Computing Center, The University of Texas at Austin, Austin, Texas*
[5]*National Institute of Standards and Technology, Boulder 80305, CO, USA*

(Dated: October 2, 2018)

I. MODELLING THE VIOLATION OF CHSH BY A POISSONIAN SOURCE OF QUBIT PAIRS

The output of a CW-pumped SPDC process can be accurately described as the emission of independent pairs distributed according to Poissonian statistics of average μ, if the time under consideration (in our case, the length of a round) is much longer than the single-photon coherence time. The goal of this Section is to provide an estimate of the observed CHSH parameter S for such a source.

The pairs being independent, it helps to think in two steps. First, each pair is converted into classical information $(\alpha, \beta) \in \{+, -\}$ with probability

$$P_Q(\alpha,\beta|x,y) = \mathrm{Tr}(\rho \Pi_\alpha^x \otimes \Pi_\beta^y)\,, \tag{1}$$

where the Π's are measurement operators. If some of the events have $\alpha = -$ ($\beta = -$), Alice's (Bob's) detector may be triggered, leading to the observed outcome $a = -1$ ($b = -1$).

For the purpose of studying CHSH, it is sufficient to consider $P(-1,+1|x,y)$ and $P(+1,-1|x,y)$, since

$$E_{xy} = 1 - P(-1,+1|x,y) - P(+1,-1|x,y)\,. \tag{2}$$

With our convention of outcomes, $P(-1,+1|x,y)$ is the probability associated with the case when Alice's detector clicks and Bob's does not. Thus, Bob's detector should not be triggered by any pair: each pair will contribute to $P(-1,+1|x,y)$ with

$$D(\alpha) \equiv P_Q(\alpha,+|x,y) + (1-\eta_B)P_Q(\alpha,-|x,y)\,. \tag{3}$$

Now, let us look at the contribution by v pairs to $P(-1,+1|x,y)$. At least one of the α's must be $-$ for the detector to be triggered; and multiple detections will also be treated as $a = -1$. Thus, a configuration in which exactly k α's are $-$ leads to $a = -1$ with probability $1-(1-\eta_A)^k$ (i.e. at least one $\alpha = -$ must trigger a detection). Obviously there can be $\binom{v}{k}$ such configurations, so the contribution of the v pair events to $P(-1,+1|x,y)$ is

$$D_v = \sum_{k=1}^{v} \binom{v}{k} [1-(1-\eta_A)^k] D(-)^k D(+)^{v-k}\,. \tag{4}$$

Finally,

$$P(-1,+1|x,y) = \sum_{v=0}^{\infty} P_\mu(v) D_v\,. \tag{5}$$

The calculation of $P(+1,-1|x,y)$ is identical, with $D(\beta) \equiv P_Q(+,\beta|x,y) + (1-\eta_A)P_Q(-,\beta|x,y)$ and η_B instead of η_A in (4).

Because the quantum probabilities appear in such a convoluted way, the optimal parameters for both the state and the measurements are not the same as for the single-pair case. Upon inspection, however, the values are close, as expected from the fact that the violation is mostly contributed by the single-pair events.

The curves presented in Fig. 2 have been obtained with a slightly modified model that includes the effect of the background events. The quantum probabilities $P_Q(\alpha,\beta|x,y)$ have been computed with $\rho = |\psi\rangle\langle\psi|$ given in Equation (4) of the main article and with projective measurements, with the values of the parameters given in the main text.

II. PROTOCOL AND SECURITY PROOF

For completeness, we will present the protocol studied in [26] and give explicit constants in its security proof. We also refer to this paper for basic definitions of (smooth) min-entropy and related quantities. It will be more convenient for us to switch to to the notation $a, b \in \{0,1\}$ for the outcome labels (instead of $a, b \in \{+1,-1\}$ of the main text) and use the language of nonlocal games with winning condition

$$w_{\mathrm{CHSH}}(a,b,x,y) = \begin{cases} 1 & \text{if } a \oplus b = x\cdot y\,, \\ 0 & \text{otherwise}\,. \end{cases} \tag{6}$$

The game winning probability is then $w = 1/2 + S/8$ in terms of the CHSH value. The optimal classical winning strategy achieve a winning probability of 0.75, while the optimal quantum strategy achieves a winning probability of $(2+\sqrt{2})/4 \approx 0.85$.

A randomness expansion protocol is a procedure that consumes r-bits of randomness and generates m-bits of almost uniform randomness. Formally, a (ϵ_c, ϵ_s)-secure

$r \to m$ randomness expansion protocol if given r uniformly random bits,

- (Soundness) For any implementation of the device it either aborts or returns an m-bit string $Z \in \{0,1\}^m$ with

$$(1 - \Pr[\text{abort}]) \left\|\rho_{ZRE} - \rho_{U_m} \otimes \rho_{U_r} \otimes \rho_E\right\|_1 \le \epsilon_\text{s}\,,$$

  where R is the input randomness register, E is the adversary system, and ρ_{U_m}, ρ_{U_r} are the completely mixed states on appropriate registers.

- (Completeness) There exists an honest implementation with $\Pr[\text{abort}] \le \epsilon_\text{c}$.

We remark that this security definition is a composable definition assuming quantum adversary, but not composable assuming a no-signalling adversary [26]. Composability allows the randomness generated to be safely used inside a larger protocol, such as quantum key distribution, without compromising the latter's security.

For a concrete randomness expansion protocol, we present the protocol studied in [26]. The protocol takes parameters γ expected fraction (marginal probability) of test rounds, ω_exp expected winning probability for an honest (perhaps noisy) implementation, and δ_est width of the statistical confidence interval for the estimation test. In an execution, for every round $i \in \{1, \dots, n\}$:

- Bob chooses a random bit $T_i \in \{0,1\}$ such that $\Pr(T_i = 1) = \gamma$ using the interval algorithm [28].
- If $T_i = 0$ (randomness generation), Alice and Bob choose deterministically $(X_i, Y_i) = (0,0)$, otherwise $T_i = 1$ (test round) they choose uniformly random inputs (X_i, Y_i).
- Alice and Bob use the physical devices with the said inputs (X_i, Y_i) and record their outputs (A_i, B_i).
- If $T_i = 1$, they compute

$$C_i = w_\text{CHSH}(A_i, B_i, X_i, Y_i)\,. \tag{7}$$

They abort the protocol if $\sum_j C_j < (\omega_\text{exp}\gamma - \delta_\text{est})n$ where j is the index of test rounds, otherwise they return $\text{Ext}(\mathbf{AB}, \mathbf{Z})$ where Ext is a randomness extractor, $\mathbf{AB} = A_1B_1...A_nB_n$ and $\mathbf{Z}$ is a uniformly random seed.

More precisely, we use a Trevisan extractor in [27] based on polynomial hashing with block weak design, because of its efficiency in terms of seed length. This is a function $\text{Ext} : \{0,1\}^{2n} \times \{0,1\}^d \to \{0,1\}^m$ such that if $H_\text{min}(\mathbf{AB}|E) \ge 4\log\frac{1}{\epsilon_1} + 6 + m$ then

$$\frac{1}{2}\left\|\rho_{\text{Ext}(\mathbf{AB},\mathbf{Z})\mathbf{Z}E} - \rho_{U_m} \otimes \rho_{U_d} \otimes \rho_E\right\|_1 \le m\epsilon_1 \tag{8}$$

The seed length of this extractor is $d = a(2\ell)^2$ where

$$a = \left\lceil \frac{\log(m - 2e) - \log(2\ell - 2e)}{\log(2e) - \log(2e-1)} \right\rceil \tag{9}$$

$$\ell = \left\lceil \log 2n + 2\log\frac{2}{\epsilon_1} \right\rceil \tag{10}$$

Now it can be shown that the entropy accumulation protocol gives the completeness and soundness of our randomness expansion protocol. However, let us mention how the input randomness affects the soundness and completeness of the final protocol.

In the protocol we assume access to a certain uniform randomness source, from which the random bits required in the protocol are generated: the T_i, X_i and Y_i. In certain rounds, X_i and Y_i are either deterministic or fully random bits and can be directly obtained from the source. On the other hand, T_i must be simulated from the uniform source (except when $\gamma = 1/2$ which is not usually the case in practice). This can be done efficiently by the interval algorithm [28]: the expected number of random bits needed to generate one Bernoulli(γ) is at most $h(\gamma) + 2$ and the maximum number of random bits needed is at most $L_\text{max} := \max\{\log\gamma^{-1}, \log(1-\gamma)^{-1}\}$. Then Lemma 16 of [26] gives us: let $\gamma > 0$, for any n there is an efficient procedure that given (at most) $6h(\gamma)n$ uniformly random bits either it aborts with probability at most $\epsilon_\text{SA} = \exp(-18h(\gamma)^3 n/L_\text{max})$ or outputs n bits $T_1, \dots, T_n$ whose distribution is within statistical distance at most ϵ_SA of n i.i.d. Bernoulli(γ) random variables. This raises both the completeness and soundness parameter of the final protocol by ϵ_SA.

In an honest implementation of the protocol, Alice and Bob execute the protocol with a device that performs i.i.d. measurements on a tensor product state resulting in an expected winning probability ω_exp. Here Lemma 8 of [26] bounds the probability of aborting using Hoeffding's inequality. That is, the probability that our randomness expansion protocol aborts for an honest implementation is

$$\Pr[\text{abort}] \le \exp(-2n\delta_\text{est}^2) =: \epsilon_\text{est}\,. \tag{11}$$

Therefore, the total completeness is bounded by $\epsilon_\text{SA} + \epsilon_\text{est}$. (Note that ϵ_est is actually the completeness parameter of the entropy accumulation protocol in [26].)

For the soundness, Corollary 11 of [26] ensures that for any $\epsilon_\text{EA}, \epsilon' \in (0,1)$ either the protocol aborts with probability greater than $1 - \epsilon_\text{EA}$ or

$$H_\text{min}^{\epsilon'}(\mathbf{AB}|\mathbf{XYT}E)_{\rho_{|\text{pass}}} > n \cdot \eta_\text{opt}(\epsilon', \epsilon_\text{EA})\,. \tag{12}$$

Together with our extractor, for all $\epsilon_1 \in (0,1)$, if the length m of the final string satisfies

$$n \cdot \eta_\text{opt}(\epsilon', \epsilon_\text{EA}) = 4\log\frac{1}{\epsilon_1} + 6 + m \tag{13}$$

then we are guaranteed that

$$\frac{1}{2}\left\|\rho_{SRE} - \rho_{U_m} \otimes \rho_{U_R} \otimes \rho_E\right\|_1 \le \epsilon'/2 + m\epsilon_1\,. \tag{14}$$

Here $\eta_\text{opt}(\epsilon', \epsilon_\text{EA})$ is given by the following equations: for h the binary entropy and $\gamma, p(1) \in (0,1]$

$$\eta_{\mathrm{opt}}(\epsilon', \epsilon_{\mathrm{EA}}) = \max_{\frac{3}{4} < \frac{p_t(1)}{\gamma} < \frac{2+\sqrt{2}}{4}} \eta(\omega_{\exp}\gamma - \delta_{\mathrm{est}}, p_t, \epsilon', \epsilon_{\mathrm{EA}}) \,, \tag{15}$$

$$\eta(p, p_t, \epsilon', \epsilon_{\mathrm{EA}}) = f_{\min}(p, p_t) - \frac{1}{\sqrt{n}} 2 \left(\log 13 + \frac{\mathrm{d}}{\mathrm{d}p(1)} g(p)|_{p_t} \right) \sqrt{1 - 2\log(\epsilon' \epsilon_{\mathrm{EA}})} \,, \tag{16}$$

$$f_{\min}(p, p_t) = \begin{cases} g(p) & \text{if } p(1) \le p_t(1) \,, \\ \frac{\mathrm{d}}{\mathrm{d}p(1)} g(p)|_{p_t} \cdot p(1) + \left(g(p_t) - \frac{\mathrm{d}}{\mathrm{d}p(1)} g(p)|_{p_t} \cdot p_t(1) \right) & \text{if } p(1) > p_t(1) \end{cases} \tag{17}$$

$$g(p) = \begin{cases} 1 - h\left(\frac{1}{2} + \frac{1}{2}\sqrt{16 \frac{p(1)}{\gamma}\left(\frac{p(1)}{\gamma} - 1\right) + 3} \right) & \text{if } \frac{p(1)}{\gamma} \in \left[0, \frac{2+\sqrt{2}}{4}\right] \\ 1 & \text{if } \frac{p(1)}{\gamma} \in \left[\frac{2+\sqrt{2}}{4}, 1\right] \end{cases} \tag{18}$$

Combined with the input sampling soundness, the total soundness is bounded by $\epsilon_{\mathrm{SA}} + \epsilon_{\mathrm{EA}} + \epsilon'/2 + m\epsilon_1$.

Parameter	Definition
ϵ_{c}	completeness, bounding honest abort probability
ϵ_{s}	soundness, bounding randomness security
ϵ_{SA}	input sampling error tolerance
ϵ'	smoothing parameter
ϵ_1	1-bit extractor error tolerance
ϵ_{EX}	randomness extractor error tolerance
ϵ_{est}	Bell estimation error tolerance
ϵ_{EA}	soundness of entropy accumulation protocol

TABLE I: Definition of security parameters.

Finally, let us count the number of random bits consumed in the protocol. It consists of the randomness used to decide if a round is a test or generation round, the randomness used to pick the inputs in a test round, and the randomness used for the Trevisan extractor. Taking into account the finite statistical fluctuations, we need at most $6h(\gamma)n$ bits to choose between test and generation except with probability ϵ_{SA}. This results in at most $2\gamma n$ testing rounds except with probability ϵ_{SA}, which equates to $2 \times 2\gamma n$ random bits being consumed for generating the inputs for test rounds. The randomness for Trevisan extractor is d bits. (Practically, after the first run of the protocol, we can omit this amount because the extractor is a strong extractor: we can reuse the seed for next run of the protocol). Summing these up, we have consumed at most $6h(\gamma)n + 4\gamma n + d$ uniformly random bits with probability at least $1 - 2\epsilon_{\mathrm{SA}}$.

In summary, for a device with $\omega_{\exp}$, any choice of $\gamma, \epsilon_1, \epsilon', \epsilon_{\mathrm{EA}} \in (0, 1)$, and n large enough, our protocol is an $(\epsilon_{\mathrm{SA}} + \epsilon_{\mathrm{est}}, \epsilon_{\mathrm{SA}} + \epsilon_{\mathrm{EA}} + \epsilon'/2 + m\epsilon_1)$-secure $[6h(\gamma)n + 4\gamma n + d] \to m$ randomness expansion protocol. That is either our protocol abort with probability greater than $1 - \epsilon_{\mathrm{EA}}$, or it produces a string of length m such that $\frac{1}{2}\left\|\rho_{SRE} - \rho_{U_m} \otimes \rho_{U_R} \otimes \rho_E\right\| \le \epsilon_{\mathrm{SA}} + \epsilon'/2 + m\epsilon_1$. The protocol consume at most $6h(\gamma)n + 4\gamma n + d$ uniformly random bits with probability at least $1 - 2\epsilon_{\mathrm{SA}}$.

III. INPUT-OUTPUT RANDOMNESS ANALYSIS

The previous Section gives a complete picture of the (one-shot) behavior of our randomness expansion protocol. For the purpose of this paper, it suffices to obtain rough estimates on the randomness output, but further optimization can be done.

For simplicity, we introduce some bounds on the resources. Since $m \le 2n$ we can let $m\epsilon_1 \le 2n\epsilon_1 =: \epsilon_{\mathrm{EX}}$ which gives $\epsilon_1 = \epsilon_{\mathrm{EX}}/(2n)$. Plugging this back in (13) gives us the number of random bits one can extract,

$$m = n \cdot \eta_{\mathrm{opt}}(\epsilon', \epsilon_{\mathrm{EA}}) - 4\log n + 4\log \epsilon_{\mathrm{EX}} - 10 \,, \tag{19}$$

for a given level of soundness $\epsilon_{\mathrm{SA}} + \epsilon_{\mathrm{EA}} + \epsilon'/2 + \epsilon_{\mathrm{EX}}$. Moreover, the protocol consumes $6h(\gamma)n + 4\gamma n + d$ bits of randomness with probability at least $1 - \epsilon_{\mathrm{SA}}$, where

$$d = a(2\ell)^2 \text{ with } a \le \frac{\log(2n - 2e) - \log(2\ell - 2e)}{\log(2e) - \log(2e - 1)} + 1$$
$$\text{and} \quad \ell \le 3\log n + 6 - 2\log \epsilon_{\mathrm{EX}} \,. \tag{20}$$

This leads to an expansion of $m - 6h(\gamma)n - 4\gamma n - d$. Hence, the output randomness rate per unit time is

$$r_n = \frac{1}{\tau}\left(\eta_{\mathrm{opt}}(\epsilon', \epsilon_{\mathrm{EA}}) - 4\frac{\log n}{n} + 4\frac{\log \epsilon_{\mathrm{EX}}}{n} - \frac{10}{n} \right) \,, \tag{21}$$

and the net randomness rate per unit time is

$$r_n^{\mathrm{net}} = \frac{1}{\tau}\left(\eta_{\mathrm{opt}}(\epsilon', \epsilon_{\mathrm{EA}}) - 4\frac{\log n}{n} + 4\frac{\log \epsilon_{\mathrm{EX}}}{n} - \frac{10}{n} - 6h(\gamma) - 4\gamma - \frac{d}{n}\right) . \tag{22}$$

These formulas are of course given for a protocol with completeness $\epsilon_{\mathrm{SA}} + \epsilon_{\mathrm{est}}$ and soundness $\epsilon_{\mathrm{SA}} + \epsilon_{\mathrm{EA}} + \epsilon'/2 + \epsilon_{\mathrm{EX}}$, where

$$\epsilon_{\mathrm{SA}} = \exp(-18h(\gamma)^3 n/L_{\max}) \tag{23}$$

$$L_{\max} = \max\{\log\gamma^{-1}, \log(1-\gamma)^{-1}\} \tag{24}$$

$$\epsilon_{\mathrm{est}} = \exp(-2n\delta_{\mathrm{est}}^2) . \tag{25}$$

The asymptotic rate for the block size $n \to \infty$ is given by taking the limit of block size n

$$r_\infty = \frac{1}{\tau}\left[1 - h\left(\frac{1}{2} + \frac{1}{2}\sqrt{\frac{S^2}{4} - 1}\right)\right] . \tag{26}$$

For the net asymptotic rate we could also take the same limit, however we could obtain a better bound by the expected behavior of the interval algorithm. Since the expected number of random bits needed to generate $T_1, ..., T_n$ is $nh(\gamma) + 2$ by [28], and γn of which is expected to be test rounds each consuming 2 random bits, we have the asymptotic net rate

$$r_\infty^{\mathrm{net}} = \frac{1}{\tau}\left[1 - h\left(\frac{1}{2} + \frac{1}{2}\sqrt{\frac{S^2}{4} - 1}\right) - h(\gamma) - 2\gamma\right] . \tag{27}$$

From an end-user perspective, one may argue that the only parameters of interest are the completeness and soundness security parameters which will constrain the rest of protocol parameters—$\gamma, \delta_{\mathrm{est}}, n, \epsilon$'s—for a given objective such as maximizing randomness rate or net randomness rate. For the illustrative plots we set the constraints

$$\epsilon_{\mathrm{SA}} + \epsilon_{\mathrm{EA}} + \epsilon'/2 + \epsilon_{\mathrm{EX}} = \epsilon_{\mathrm{s}} , \tag{28}$$

$$\epsilon_{\mathrm{SA}} + \epsilon_{\mathrm{est}} = \epsilon_{\mathrm{c}} , \tag{29}$$

and fix $\epsilon_{\mathrm{c}} = 10^{-10}, \epsilon_{\mathrm{s}} = 10^{-10}$. One then maximizes randomness output or net randomness output given these constraints. This gives us the best (as measured by our objective function) protocol parameters within the relaxations made to obtain (19) and (20).

However, in the main text we take a simpler approach without optimizing over the variables $\gamma, \delta_{\mathrm{est}}, \epsilon$'s. For each block size n, we compute ϵ_{SA} as given by (23) which then fixes δ_{est} via $\epsilon_{\mathrm{est}} = 10^{-10} - \epsilon_{\mathrm{SA}}$ and (25). The remaining ϵ's which have weight $10^{-10} - \epsilon_{\mathrm{SA}}$ are chosen in a $1 : 2 : 1$ ratio of $\epsilon_{\mathrm{EA}} : \epsilon' : \epsilon_{\mathrm{EX}}$, which is guaranteed to add up to the specified level of completeness and soundness. This approach is not far from optimal in the regime of large block size n. This results in the experimental points reported in Figure 3 in the main text.

IV. RANDOM BITS EXTRACTION PROCEDURE

As mentioned in the main text, the data observed during the experiment contains certified randomness. Here we describe the procedure we use to extract this randomness in a finite run of the experiment. We consider two blocks of data, corresponding to an acquisition time of ≈ 42.8 min (`dataset1`) and ≈ 17.33 hours (`dataset2`).

The randomness protocol we use (described in Section II) relies on two elements:

- an honest implementation, and
- security parameters.

These elements must be chosen adequately before proceeding to the extraction. Indeed, if a too optimistic honest implementation is chosen, for instance, the data observed will fail to pass the test, and the whole protocol will abort: no randomness can then be extracted.

Moreover, our setup allows us to choose freely the bin width which can significantly affect the amount of certified randomness.

We dedicate a fraction γ_{calib} of our data to the estimation of these parameters so as to maximize the amount of randomness certified. The randomness protocol is then run with these parameters on the remaining fraction $(1-\gamma_{\mathrm{calib}})$ of the data only. We determine the fraction of data γ_{calib} to use for the calibration of the randomness extraction procedure from a simulation of the experiment. We estimate the number of random bits that can be certified from an experiment of the envisioned length if a fraction γ_{calib} of the data is dedicated to calibration purpose (all other parameters being set as expected). We choose the value of γ_{calib} that maximizes this quantity. We find that $\gamma_{\mathrm{calib}} = 22\%$ is adequate for `dataset1`, and $\gamma_{\mathrm{calib}} = 8\%$ for `dataset2`.

We then proceed to define the parameters of the randomness protocol. The security parameters $\epsilon_{\mathrm{s}}, \epsilon_{\mathrm{c}}$ are set a priori, with all the other parameters derived as described in Sections II and III, with $\gamma = 1$. We define *honest implementation* an implementation which reproduces the CHSH violation observed during the calibration stage with probability

$$P(S_{\mathrm{exp}} \geq S_{\mathrm{calib}}) \geq \epsilon_{\mathrm{calib}} . \tag{30}$$

For concreteness, we set $\epsilon_{\mathrm{calib}} = 10^{-10}$. This step guarantees that we will not overestimate the amount of Bell violation which we can expect from a honest experiment. This is crucial for the whole certification procedure to succeed with a large probability. We then have

$$w_{\mathrm{exp}} = w_{\mathrm{calib}} - \delta_{\mathrm{calib}} , \tag{31}$$

with $\delta_{\text{calib}} = B(\omega_{\text{exp}}, (1-\gamma_{\text{calib}})n, \omega_{\text{calib}})$, where

$$B(p,n,q) = \sum_{i=0}^{nq} \binom{n}{i} p^i (1-p)^{n-i} \qquad (32)$$

is the cumulative distribution of n Bernoulli variable with parameter p. For simplicity, we use the upper bound

$$\delta_{\text{calib}} \leq \sqrt{\frac{\log(1/\epsilon_{\text{calib}})}{2n}} \qquad (33)$$

valid for all winning probability ω_{calib}, which leads to a conservative estimate of the honest implementation Bell violation S_{est}.

Having fixed all security parameters and defined our honest implementation, we are now left with the choice of the bin width. For this, following the procedure discussed in the main text, we compute the number of certified random bits that we can hope to certify in the remaining $(1-\gamma_{\text{calib}})$ fraction of the data as a function of the bin width. We then choose the bin width which yields the maximum rate of random bits. We find that optimal bin widths 8.9 μs for `dataset1` and 5.35 μs for `dataset2`. This allows us to define how the remaining data is to be treated: first, we extract the outcomes corresponding to the chosen bin width, then we use the exact number of bins so extracted to compute precisely the threshold Bell violation $w_{\text{exp}} - \delta_{\text{est}}$ and the number of certified bits m corresponding to this dataset, finally we check whether the data indeed yields a Bell violation larger than $w_{\text{exp}} - \delta_{\text{est}}$. If this is not the case, we abort. Otherwise, we apply a randomness extractor on the string of outcomes. Both datasets pass this test.

Finally, we use the Trevisan extractor implemented by Mauerer et al. [27], and further improved by Bierhorst et al. [11] to extract the certified bits. Our modified source code is available online at `https://github.com/jdbancal/libtrevisan`. The advantage of Trevisan extractors over other kinds of randomness extractors is that they require little initial seeds, and that they are composable, strong extractors, and secure against quantum side information. Following the suggestion of [27], we use the block weak design construction with a RSH extractor to maximize the number of extracted bits. The extractor also require a supply of seed randomness. For this, we use the bits generated with the random number generator described in [31].

In the end, we extract 617 920 and 35 799 872 uniformly random bits from `dataset1` and `dataset2` respectively, using seeds of length 1 808 802 and 2 923 224. The corresponding rates, calculated including the acquisition time of the calibration data, source phase lock and basis switching, are $\approx$ 240 bits/s and $\approx$ 573 bits/s. If we consider only the time necessary for the data acquisition, we obtain net randomness rates of $\approx$ 396 bit/s and $\approx$ 943 bit/s.

These rates do not include the processing time of the Trevisan extractor. This classical computation took 9 hours for `dataset1` on a machine processing 24 threads in parallel. The bits extracted can be found in the Supplementary Material as well (see file `extracted_randomness-dataset1.bin`). For `dataset2` we estimated an unreasonable extraction time of $\approx$ 18 years if run on the same hardware. In order to reduced this time to few days it would be necessary to employ thousands of cores. We decided not to pursue this effort as we do not think this would add any more interest to the presented results.

We used the NIST Statistical Test Suite [30] to ensure the quality of generated strings is at least on par with acceptable pseudo-randomness. This suit of test can only verify the uniformity of the generated random string, it does not certify its privacy. The string generated from `dataset1` passed all the tests that are meaningful for this relatively short data sample, assuming an acceptable significance level $\alpha = 0.01$. The result of the individual tests are summarized in table II.

Test	P–value	Proportion
Frequency	0.590949	96/97
Block Frequency	0.275709	95/97
Cumulative Sums Forward	0.964295	96/97
Cumulative Sums Backward	0.637119	96/97
Runs	0.162606	97/97
Longest Run of Ones	0.590949	96/97
Discrete Fourier Transform	0.183769	96/97

TABLE II: Result of the NIST Statistical Test Suite for the bits extracted from `dataset1`. We split the random bits into 97 sequences of 6300 bits each.