

Notes on Interrogating Random Quantum Circuits

Luís T. A. N. Brandão*¹ and René Peralta*²

May 29, 2020



Abstract

Consider a quantum circuit that, when fed a constant input, produces a fixed-length random bit-string in each execution. Executing it many times yields a sample of many bit-strings that contain fresh randomness inherent to the quantum evaluation. When the circuit is freshly selected from a special class, the output distribution of strings cannot be simulated in a short amount of time by a classical (non-quantum) computer. This quantum vs. classical gap of computational efficiency enables ways of inferring that an honest sample contains quantumly generated strings, and therefore fresh randomness. This possibility, initially proposed by Aaronson, has been recently validated in a “quantum supremacy” experiment by Google, using circuits with 53 qubits.

In these notes, we consider the problem of estimating information entropy (a quantitative measure of randomness), based on the sum of “probability values” (here called QC-values) of strings output by quantum evaluation. We assume that the sample of strings, claimed to have been produced by repeated evaluation of a quantum circuit, was in fact crafted by an adversary intending to induce us into over-estimating entropy. We analyze the case of a “collisional” adversary that can over-sample and possibly take advantage of observed collisions.

For diverse false-positive and false-negative rates, we devise parameters for testing the hypothesis that the sample has at least a certain expected entropy. This enables a client to certify the presence of entropy, after a lengthy computation of the QC-values. We also explore a method for low-budget clients to compute fewer QC-values, at the cost of more computation by a server. We conclude with several questions requiring further exploration.

Keywords: certifiable randomness, distinguishability, entropy estimation, gamma distribution, public randomness, quantum randomness, randomness beacons.

*National Institute of Standards and Technology (Gaithersburg USA).
ORCID: ¹[0000-0002-4501-089X] and ²[0000-0002-2318-7563].
Opinions expressed in this paper are from the authors and are not to be construed as official or as views of the U.S. Department of Commerce. Certain commercial entities, equipment, or materials may be identified in this document in order to describe an experimental procedure or concept adequately. Such identification is not intended to imply recommendation or endorsement by NIST, nor is it intended to imply that the entities, materials, or equipment are necessarily the best available for the purpose.

Index of sections

| | |
|--|-----------|
| 1. Introduction | 2 |
| 1.1. System model | 2 |
| 1.2. Entropy of a sample | 4 |
| 1.3. Organization | 4 |
| 2. Exponential model | 5 |
| 2.1. The frequency-density representation | 5 |
| 2.2. Summary statistics | 6 |
| 2.3. Entropy per honest string | 6 |
| 2.4. Sampling with vs. without replacement | 6 |
| 3. Sums of QC-values | 7 |
| 3.1. Statistics of interest | 8 |
| 3.2. CDFs of sums of i.i.d. variables | 8 |
| 3.3. Testing honest sampling | 9 |
| 3.4. Threshold vs. probability | 9 |
| 3.5. Sample sizes vs. thresholds | 9 |
| 4. Low-budget clients | 11 |
| 4.1. Truncated QC-values | 11 |
| 4.2. Sum of truncated QC values (STQC) | 12 |
| 5. Entropy estimation | 12 |
| 5.1. Overview | 13 |
| 5.2. The client | 13 |
| 5.3. The pseudo-fidelity adversary | 14 |
| 5.4. The collisional adversary | 16 |
| 5.5. Final randomness for applications | 20 |
| 5.6. Classes of adversaries | 20 |
| 6. Concluding remarks | 21 |
| Acknowledgments | 22 |
| References | 22 |
| A. Terminology | 22 |
| A.1. Abbreviations | 22 |
| A.2. Acronyms | 23 |
| A.3. Symbols | 23 |
| B. Expected value and variance | 23 |
| B.1. Auxiliary primitives | 23 |
| B.2. Expected values | 24 |
| B.3. Variances | 24 |
| B.4. The chosen-count sampling case | 24 |
| C. Sum of QC-Values (SQCs) | 24 |
| C.1. CLT approximation | 24 |
| C.2. Exact Gamma distributions | 25 |
| C.3. A Gamma approximation | 27 |
| D. Discretization of QC-values | 28 |
| D.1. Individual probabilities | 28 |
| D.2. Collisions | 28 |
| D.3. Approximate sum of “entropies” | 30 |
| E. Tables with more detail | 32 |

List of Figures

| | | |
|-----------|--|----|
| Figure 1 | QC-values upon uniform string sampling | 5 |
| Figure 2 | QC-values upon quantum string sampling | 5 |
| Figure 3 | Various PDFs of QC-values | 6 |
| Figure 4 | Various PDFs of SQC with $m = 10$ | 8 |
| Figure 5 | PDF approximations of QC-values $X_{F,.5}$ | 9 |
| Figure 6 | PDF approximations of SQC $X_{F,m=5,.5}$ | 9 |
| Figure 7 | Inverse CDFs of SQC with $m = 10^6$ | 10 |
| Figure 8 | Inverse CDFs of SQC with $m = 10^7$ | 10 |
| Figure 9 | Sample size vs. FN=FP, with $\phi_1 = 0.002$ | 10 |
| Figure 10 | Sample size vs. FN=FP, with $\phi_1 = 0.01$ | 11 |

List of Tables

| | | |
|----------|--|----|
| Table 1 | Statistics of QC-values | 6 |
| Table 2 | Expected number c of collisions ($N = 2^{53}$) | 7 |
| Table 3 | Statistics of SQCs of m strings | 8 |
| Table 4 | Number of strings for SQC distinguishability | 11 |
| Table 5 | TQC truncation thresholds | 11 |
| Table 6 | Statistics of Truncated QCs | 12 |
| Table 7 | Number of client-verified TQC values | 12 |
| Table 8 | Number of strings for SQC distinguishability | 16 |
| Table 9 | Comparison pseudo-fidelity vs. collisional | 19 |
| Table 10 | Gamma vs. Normal (CLT) approximations | 27 |
| Table 11 | Entropy approximations: Uniform, $q = 1$ | 31 |
| Table 12 | Entropy approximations: Uniform, $q \geq 1$ | 32 |
| Table 13 | Entropy approximations: Quantum, $q \geq 1$ | 32 |
| Table 14 | Sample size for SQC distinguishability | 33 |
| Table 15 | Sample size for STQC distinguishability | 34 |
| Table 16 | Statistics per bin c and budget factor b | 35 |

1. Introduction

The recent experimental proof of *quantum supremacy* using Noisy Intermediate-Scale Quantum (NISQ) devices showed that quantum circuits with 50+ qubits can now be sampled with significant fidelity [AABB+19]. Using this technology, it is possible, in principle, to generate a sample of bit-strings that can at a later time be “certified” as containing strings that were quantumly sampled [Aar19], implying it can be externally verified that the sample contained at least a minimum of fresh entropy.

We address the following two related questions: *Under a claim that a sequence of bit-strings has been generated by sampling a given quantum circuit, how much entropy can be safely assumed to be contained in it? Given a goal of entropy, how many strings should be sampled to enable a verification with high assurance?* We consider an adversarial setting, as usual in cryptography, where the claimant tries to trick us into over-estimating entropy.

A metrological viewpoint. In the scope of the National Quantum Initiative Act [NQIA] from the U.S. Congress, the National Institute of Standards and Technology (NIST) is interested in the development of quantum computing and its applications. A potential application, within reach of current or soon-to-reach state of the art, is the production of certifiable randomness based on evaluation of random quantum circuits [Aar19]. The Computer Security Division at NIST has a special interest in the area of randomness, which has an essential role in cryptography. “Certifiable” randomness in particular may be useful in the context of *public randomness*, such as that produced by randomness beacons [KBPB19].

In these notes we take the viewpoint of a metrology body [NIST] in doing a preliminary evaluation of the parameters of a potential application for obtaining certifiable randomness. We consider a cryptographic perspective, e.g., when asking what false positive and false negative rates should be considered in distinguishability experiments. This is a preliminary analysis and should be taken as such. We do not investigate here the complexity-theoretic basis for the assumption that sampling from certain quantum circuits can be done efficiently with a quantum computer but not classically [AC16]. However, we explore how certain attacks drastically reduce entropy from the set of bit strings to be certified. The analysis thus illustrates the need to define appropriate safety margins for diverse parameters (e.g., number of strings to sample), and rules out certain ranges thereof.

1.1. System model

1.1.1. The operator

We want to compare an *honest operator* of the quantum computer — which generates a sample with fresh entropy via an honest quantum evaluation, leading a sample to be accepted with a statistically high probability (i.e., low false-negative rate) — vs. a *malicious operator* — which maliciously minimizes the amount of entropy in a sample crafted to still be acceptable with a not too-low probability (i.e., a not-too-low false-positive rate).

In any of the cases, we assume that the circuit received for quantum evaluation was unpredictable to the operator. It could for example be based on fresh public randomness, or have been provided by a client interested in the experiment. The operator then needs to output a sample of strings (supposedly by evaluation of the circuit) after a short amount of time, namely before being able to compute their output probabilities.

The honest case. The honest operator of the quantum computer repeatedly evaluates the quantum circuit, to probabilistically obtain output strings that, based on the

computational model, inherently contain fresh entropy. It is assumed that the sampling from such probability distribution in a fast way is only possible by way of said quantum computation. Later, a classical super-computer performs a lengthy computation (e.g., of a few days) of the output probabilities of the output strings (e.g., see Ref. [PGNHW19] for an analysis of the complexity of simulating probability values for circuits with 54 qubits). Finally, a statistical analysis of those probabilities confirms that some strings must have been output by quantum evaluation, and, consequently, that the sample set contains entropy that was fresh at the time of the sample generation. Such randomness is then denoted as “certified” (or “certifiable”) randomness.

The adversarial case. In adversarial contexts (as with cryptographic applications), we are interested in scenarios where the operator of the quantum computer wants to trick us into accepting a maliciously produced sample. Therefore, we consider that the sampling may have been performed in a variety of ways, such as:

- uniformly at random from a defined set S of strings;
- as a pseudo-randomly generated output computed from a fixed secret seed;
- using rejection sampling on the output of the circuit;
- a mix of the above and other unknown methods.

The malicious goal is to minimize the entropy of the sample, while having a not-too-low probability of it being a posteriori accepted by the statistical test performed by a client, who will compute and take in consideration the probability values of the strings in the sample. We consider concrete specifications of adversary in Section 5.

The quantum computation. The operator can use a quantum computer to quantumly-evaluate circuits that output strings of n bits. We denote by $S_n = \{0,1\}^n$ the set of n -bit strings. There are $\#(S_n) \equiv N = 2^n$ such strings. The honest computer implementation is characterized by a *fidelity* parameter ϕ . The sampling is with probability ϕ from correct quantum evaluation of the circuit, and otherwise (i.e., with probability $1 - \phi$) uniform from S_n . As of this writing, both Google and IBM report having quantum computers with more than 50 bits. As a reference in this work, we use the specification reported about the quantum computer of Google, which can evaluate circuits with 53 qubits at a fidelity of about 0.002 [AABB+19].

When honestly evaluating the quantum circuit, with fidelity ϕ , the computer operator is not able to distinguish whether an output string was obtained by uniform selection (with probability $1 - \phi$) or by a correct circuit evaluation (with probability ϕ). A malicious operator can naturally decide to sample strings in arbitrary ways, but (in the considered model) cannot determine, before

an expensive and lengthy classical computation, anything new about the probability that a given string would have been output from a correct circuit evaluation.

1.1.2. Circuits and probabilities

The circuits, with particular specifications [AABB+19, Fig. 3], are selected from a class with large cardinality, such that it is infeasible to precompute useful information about a non-negligible proportion of circuits.

QC-values. Each quantum circuit \mathcal{C} has its own probabilistic distribution of output strings upon quantum evaluation. We are interested in the set

$$\text{QCVALUES} = \{\text{Prob}(s \leftarrow \mathcal{C}) : s \in S_n\} \quad (1)$$

of “probabilities of occurrence” — here denoted as “QC-values” — of the output strings. For simplicity we used a set, assuming all probabilities are different; otherwise we could describe a list with possible repetitions.

Assumptions. The subsequent analysis in these notes is based on assumptions whose coverage we have not independently investigated. (Different assumptions may invalidate some of our estimates of security or entropy in adversarial settings.) An important notion is what we call a “short amount of time” — the time duration between the moment the adversary learns the circuit specification and the *deadline* for publishing a sample of output strings. At a high level, the assumptions are:

1. for all circuits in the class, the QC-values of the output strings fit an exponential model: the density of QC-values (real numbers between 0 and 1) is assumed well approximated by an “exponential” curve. More concretely, the “frequency density” is a normalized negative exponential function $f(p) = N \cdot e^{-N \cdot p}$ of the “probability value” p [BISB+18].
2. a quantum computer can efficiently evaluate the circuit many times within a *short* amount of time;
3. without prior knowledge of (an approximation of) the probability values, classical computers cannot, within a *short* amount of time, simulate a circuit evaluation with the appropriate output distribution;
4. a computer (quantum or classical) cannot, within a *short* amount of time, compute a useful approximation of probability values of the output strings;
5. a classical super-computer can calculate the QC-values (i.e., $\text{Prob}(s \leftarrow \mathcal{C})$ for any string s) after a *moderately large* amount of time, at a large-but-possible-in-practice computational cost.

The abilities and inabilities mentioned above depend on the number n of qubits. In these notes we focus on

$n = 53$ qubits. We assume the computations referred in Assumptions 3, 4 and 5 require resources exponential in n . Thus, n needs to be chosen such that the exponential cost is infeasible in a *short* amount of time, but feasible in a *moderately large* amount of time.

Towards certified randomness. Relying on the above, and based on a proposal by Aaronson [Aar19], here is, at a high level, a potential experiment to produce certified (i.e., externally verifiable) randomness:

1. The operator is given the specification of a quantum circuit freshly chosen at random from a given class. (For example, in the context of public randomness, the choice may be based on a timely output of a trusted randomness beacon.)
2. Soon thereafter, the operator evaluates the circuit many times and publishes the output strings.
3. Later, a classical supercomputer computes the “QC-values” corresponding to those sampled strings.
4. By statistically analyzing the “QC-values”, one then gains assurance (or not) that at least some strings were quantumly produced and thus have entropy.

For efficiency of execution we consider a sampling of many strings from the same circuit. Comparatively, the proposal by Aaronson uses one new circuit for each string, to enable, with respect to entropy estimation, a security reduction to a complexity theoretic hardness assumption. It is an open problem what kind of reduction can be made for the case of sampling multiple strings from a single circuit. Section 2.4 considers tradeoffs between the two approaches, and mentions the possible intermediate case of several circuits with several strings each.

Random variables. For the initial statistical analysis in these notes, the main random variable of interest for each sampling experiment is the sum, across sampled strings, of the QC-values. Recall that these are the “probability values” that a correct evaluation of the quantum circuit — what we denote as *quantum sampling* fidelity 1 — would output such strings. We denote by X the random variable corresponding to this sum of QC-values (SQC). We use indices to indicate the type of sampling (U, Q, F, C) and its parameters (m, q, ϕ):

- **Uniform:** $X_{U,m}$ (m strings are sampled uniformly)
- **Pure Quantum:** $X_{Q,m}$ (m strings are obtained by correct quantum evaluation of the circuit)
- **Fidelity ϕ :** $X_{F,m,\phi}$ (each of m strings is obtained either, with probability ϕ , by correct quantum evaluation, or, with probability $1 - \phi$, uniformly)
- **Chosen-count q :** $X_{C,m,q}$ (q strings are sampled by correct quantum evaluation, and the other $m - q$ are pseudo-randomly selected)

The index m may be omitted when it is 1. Note that the Pure Quantum and the Chosen-count sampling are only possible with a quantum computer with fidelity 1.

Distinguishability. In these notes, we are focused on the problem of distinguishing honest Fidelity sampling from a quantum circuit vs. malicious sampling performed by an adversary with the goal of inducing over-estimation of the entropy in the sample. We propose parameters for sampling experiments, distinguishability thresholds, and entropy estimation, assuming an adversarial setting.

1.2. Entropy of a sample

The meaning of entropy can be elusive and subject to nuances. The measure of entropy of a string, or of a sequence of strings, only makes sense with respect to a probability distribution of outputs. For example, in the case of a uniform distribution over the set of n -bit strings we say that each occurring string has n -bits of entropy. Conversely, a string obtained pseudo-randomly from a fixed a priori determined seed (whose bits are not counted) has overall entropy 0.

Typical interpretations of entropy relate to unpredictability, compressibility, and/or reproducibility. We focus our analysis on estimating Shannon entropy (the expected negative binary logarithm, $-\log_2$, of probabilities) rather than minimum entropy ($\min(-\log_2)$).

Estimating entropy. In these notes, we estimate the entropy of a sample viewed as a vector of strings. We consider an adversarial setting where each string can be obtained from a different probability distribution. The distributions can have dependencies, such as those related to sampling without replacement, and/or from sorting the sequence based on some order relation. Even though the adversary has an incentive to minimize entropy, that goal is conditioned on the sample being accepted by the client with a certain minimum probability. For appropriately parametrized experiments, this requires the adversary to use some quantumly-obtained strings that have associated entropy. If we take into account the conditional form of probability distributions, then we can consider the overall entropy of the sample as sum of “entropies” of its consecutive strings, when for each string there is an underlying conditional probability distribution that takes into account the dependency on the previous strings. In fact, we will consider an adversary that selects strings one by one, with dependencies across each other, and argue that such adversary is optimal (within stated constraints).

1.3. Organization

Section 2 analyzes the exponential model of QC-values. Section 3 discusses the distribution and statistics of “Sums of QC-values” (SQCs) for several sampling experiments, and determines optimal thresholds for distin-

guishability. Section 4 explores alternative parameters for settings where the client has a “low budget” for verifying QC-values computed by a distrusted server. Section 5 considers the estimation of entropy in the face of an adversarial sampling. Section 6 concludes with suggested questions for followup. The Appendix contains auxiliary details. Section A defines abbreviations, acronyms and symbols. Section B derives formulas for the expected value and variance of several distributions. Section C analyzes several distributions of sums of QC-values. Section D considers the discretization of QC-values and the statistics when in the face of collisions. Section E presents additional large tables.

2. Exponential model

2.1. The frequency-density representation

We consider the model [BISB+18] where the quantum circuit outputs strings whose *frequency density* (f , a continuous approximation) of QC-values is defined by an exponential distribution (Exp) with rate N :

$$f(p) = \text{Exp}[N] = N \cdot e^{-N \cdot p} \quad (2)$$

2.1.1. U: Uniform sampling

The function f is not a probability density function (PDF) of the output strings, but rather a PDF of their QC-values when strings are sampled uniformly from the set S_n of all n -bit strings. The corresponding cumulative distribution function (CDF) F is

$$F(p) = 1 - e^{-N \cdot p}. \quad (3)$$

In the continuous model we calculate statistics while integrating between 0 and infinity, but the contribution between 1 and infinity is negligible ($\sim e^{-N}$). The actual discrete QC-values (see a discretization in Appendix D), being probabilities, are between 0 and 1.

Figure 1 plots the frequency density curve (f), and its accumulator (integral, times N), along with a histogram of f . In the histogram, the height of each constant-width bin equals N times the integral of the curve between the limits of the bin. For example: 63.2 % is the fraction of strings with “QC-values” $\leq 1/N$. The scales of the axes are represented relative to $N = 2^n$. The curve vanishes exponentially fast.

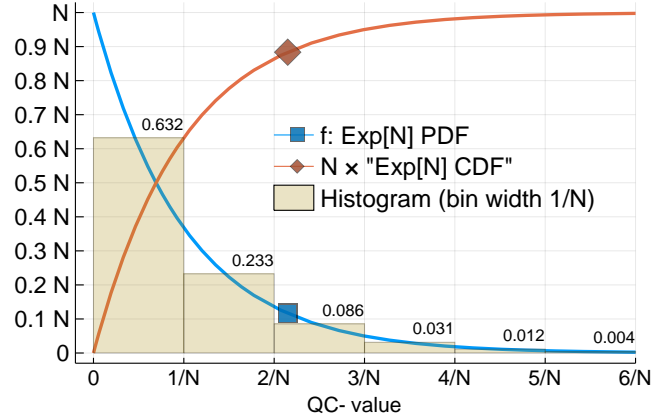


Figure 1: QC-values upon uniform string sampling

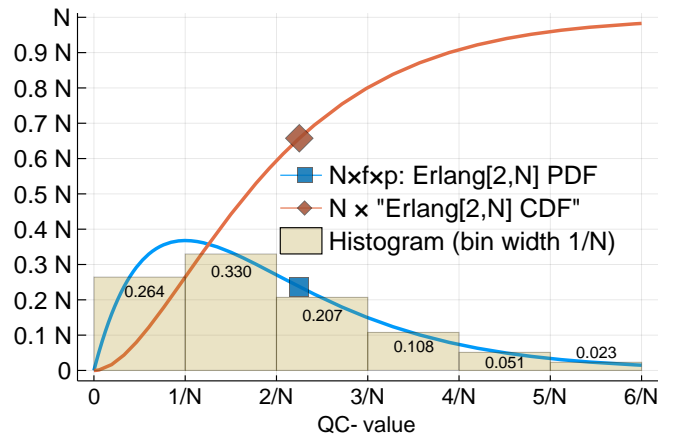


Figure 2: QC-values upon quantum string sampling

2.1.2. Q: [pure] Quantum sampling

For more insight, Fig. 2 plots the density curve of frequency times QC-value, and its accumulator (times N). Since QC-values are the probabilities of QC-values upon sampling by quantum circuit evaluation, the corresponding PDF (f_Q) and CDF (F_Q) of QC-values are as follows:

$$f_Q(p) = N \cdot f(p) \cdot p = N^2 \cdot p \cdot e^{-N \cdot p} \quad (4)$$

$$F_Q(p) = 1 - (N \cdot p + 1) \cdot e^{-N \cdot p}. \quad (5)$$

In Fig. 2, the accumulator curve shows $F_Q(p)$ multiplied by N to enable simultaneous view with $f_Q(p)$.

This is called an Erlang distribution, with “shape” 2 and “rate” N . Interestingly, this random variable (X_Q) corresponds to the sum of two exponential random variables (X_U) with rate N . This stems from the additivity of the Erlang distribution, of which the exponential distribution is the special case with shape 1. (It would be interesting to explore whether this equivalence as a sum of two independent exponential variables may have a more insightful interpretation as a quantum phenomenon.)

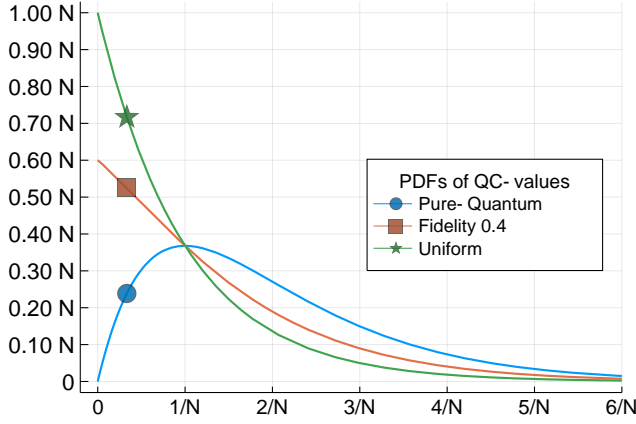


Figure 3: Various PDFs of QC-values

2.1.3. F: Fidelity sampling (the practical case)

In practice, honest sampling uses a quantum computer characterized by a fidelity ϕ between 0 and 1. This fidelity is the probability that during the quantum computation all gates in the circuit function without fault. When one or more gates fail, the model assumes that the evaluation yields a uniformly random bit-string. The resulting PDF and CDF of QC-values are a mix of the uniform and the pure-quantum case, as follows:

$$f_\phi(p) = (1 - \phi) \cdot f(p) + \phi \cdot f_Q(p) \quad (6)$$

$$F_\phi(p) = (1 - \phi) \cdot F(p) + \phi \cdot F_Q(p) \quad (7)$$

The fidelity case (allowing a generic ϕ) generalizes both the uniform ($\phi = 0$) and the pure quantum cases ($\phi = 1$). Figure 3 plots at once three PDFs of QC-values.

2.2. Summary statistics

From the PDFs of QC-values for each type of sampling (e.g., U , Q , F), we can derive statistics of interest, such as the expected value (E) and variance (V). Table 1 shows the resulting formulas. The calculations are detailed in Appendix B. Two observations:

- Both E and V in the pure quantum case are twice the corresponding statistic of the uniform case.
- The less trivial result is the variance $V[X_{F,\phi}]$ for

Table 1: Statistics of QC-values

| Sampling type | Random variable | Expected value $E(X)$ | Variance $V(X)$ |
|---------------|-----------------|-----------------------|-----------------------------------|
| Uniform | X_U | $1/N$ | $1/N^2$ |
| Pure Quantum | X_Q | $2/N$ | $2/N^2$ |
| Fidelity | $X_{F,\phi}$ | $(1 + \phi)/N$ | $(1 + \phi \cdot (2 - \phi))/N^2$ |

fidelity sampling, since for each individual sampled string the possible outcome as a uniform sampled string (X_U) is not independent of the possible outcome as a correct quantum circuit output (as X_Q).

2.3. Entropy per honest string

As a reference case, consider the expected entropy of an individual string quantumly-sampled with fidelity 1. Such entropy is slightly less than the number $n = 53$ of bits per string, since the quantumly-generated strings do not have a uniform distribution. From the exponential model for the frequency density of QC-values, we can in a first approximation consider a notion of differential entropy (using log base 2):

$$h = \int_{p=0}^{\infty} N \cdot f(p) \cdot p \cdot \log_2(p) \cdot dp \quad (8)$$

In the integral, the factor $N \cdot f(p)$ corresponds to the density-number of strings that have probability p . The approximate result (ignoring terms negligible in n) is

$$h = \log_2(N) + (\gamma - 1)/\log(2) = n - 0.60995, \quad (9)$$

where $\gamma \approx 0.57722$ is the Euler-Mascheroni constant. For $n = 53$ this means about 52.39 bits of expected entropy per string. This is the continuous approximation of the Shannon entropy (10), which sums, across every string, the product of each discrete QC-value and its \log_2 . Appendix D.1 considers a discretization.

$$h = \sum_{i=1}^N p_i \cdot \log_2(p_i). \quad (10)$$

The 52.39 bits of entropy per string are valid in the setting of honest fidelity-1 evaluation with replacement. That value changes when considering a sample composed of strings required to be distinct, and even more so if they may be selected adversarially. Appendix D.2 considers an adversary that outputs a sample only after observing the result of many quantum evaluations of the circuit, possibly observing repeated outputs.

2.4. Sampling with vs. without replacement

Independence vs. collisions. When sampling with replacement, the probability of string collisions becomes more significant as the sample size increases. When uniformly sampling with replacement from a set with N elements, the collision probability of about 50 % occurs when the sample size m is about $\sqrt{2 \cdot \log(2) \cdot N}$. For $N = 2^{53}$ this corresponds to about 111.7 million, i.e., $m \approx 0.776 \cdot 2^{26.5}$ strings. For non-uniform distributions, such as for quantum string sampling, collisions are expected to start earlier and be more frequent.

Table 2 shows a few examples: the expected number of collisions is 1 when $m \approx 134.2 \cdot 10^6$ for uniform sampling, or when $m \approx 94.9 \cdot 10^6$ for quantumly sampling with fidelity 1; if fixing the sample size to 2^{32} strings, then the expected number of collisions is about 2^{10} for uniform sampling, and about 2^{11} for fidelity-1 quantum sampling.

Table 2: Expected number c of collisions ($N = 2^{53}$)

| | $E(\#\text{coll})$ | m |
|---------|--------------------|-------------------------------------|
| Uniform | 1 | $2^{27.0} \approx 134.2 \cdot 10^6$ |
| Quantum | | $2^{26.5} \approx 94.9 \cdot 10^6$ |
| Uniform | $\approx 2^{10}$ | 2^{32} |
| Quantum | $\approx 2^{11}$ | |

Explicit removal of collisions. We require that the final sample does not contain collisions, (i.e., repeated strings marked as output of the same circuit). In the honest case this equates to sampling *without replacement*. Thus, we require that the client rejects any sample containing any pair of equal strings claimed to have been generated from the same circuit. If a client does not check for collisions, then an adversary could simply produce a sample as a sequence of m copies of a pseudo-randomly generated string (with 0 entropy), and have a noticeable probability of having an average probability value as high or larger than an honest string from quantum evaluation.

Despite the mentioned requirement, as an approximation we calculate statistics and thresholds while assuming a sampling *with replacement*. This is valid when the string length is sufficiently large. It is worth noticing an (impractical) extreme case where the approximation would not hold: sampling *without replacement* exactly $N = 2^n$ strings, from a single circuit with n qubits, yields 0 entropy (since all possible strings are present), apart from the entropy contained in the ordering of the strings (which can be 0 if maliciously ordered).

Changing the circuit. A repetition is counted as a collision only when it happens within the same circuit. Thus, collisions in the final sample can be inherently avoided if requiring each string to be associated with a different circuit. However, there is an efficiency motivation for proposing sampling from a single circuit or only a few circuits, assuming that: (i) it is more efficient to reevaluate a circuit, compared to preparing a new circuit for first-time evaluation; (ii) it is more efficient to compute many QC-values for a single circuit, compared to a single QC-value for each of many circuits. If the sample size is large enough to be reasonable to approximate the sampling as being with replacement, then the statistics of QC-values are similar between the single circuit and many circuits cases.

An efficiency tradeoff. An alternative option to reduce the probability of collisions, while not substantially decreasing efficiency, is to partition the sampling across several circuits. Suppose the time taken to prepare a new circuit for first time sampling is 10^4 times longer than the time it takes to repeat one evaluation (e.g., 10 ms vs. 1 μ). Then, allowing a sampling of up to 10^5 strings per circuit would: (i) speed up the evaluation by about 9091 times, compared to the case of one string per circuit; while (ii) only incurring a time increase factor of about 10 % compared to the case of sampling all strings from the same circuit. With respect to verification time, the complexity of verifying strings across several circuits increases with the number of circuits, if assuming that the computation/verification of several strings within each circuit increases sub-linearly with the number of strings (e.g., that verifying ten QC-values is less than 10 times costlier than verifying a single QC-value).

A gap tradeoff. The range of possibilities between one and many circuits may also allow tuning the gap between the time to sample strings and the time to compute all QC-values. Compared with the many-string-from-a-single-circuit setting, increasing the number of circuits to simulate may substantially increase the time for computing QC-values, without significantly increasing the time to evaluate the corresponding strings.

A subtle adversarial issue. An adversary who is able to perform a very fast repeated evaluation of a quantum circuit could possibly produce many string collisions. The analysis of the frequency of collisions for each string would show which strings are likely to have higher QC-values, and could thus provide to the adversary an advantage in skewing the SQC statistic, for example to adversarially affect an estimation of entropy. This effect can be significant if the number of qubits is small, namely when the number of possible strings is smaller than the number of evaluations the adversary is able to perform within the time window to publish a sample.

3. Sums of QC-values

In this section we consider the distribution of the **Sum** of **QC-values** (SQC). This distribution relates to the Heavy Output Generation (HOG) test [AC16], where one wants to find whether the SQC of generated outputs is heavy enough to be reasonable to accept that some of the originating strings must have been quantumly obtained.

We denote by $X_{F,m,\phi}$ the random variable SQC in the honest case where m strings are sampled by a quantum computation with fidelity ϕ . The cases of **uniform** sampling ($X_{U,m}$) and **pure-quantum** sampling ($X_{Q,m}$) are special cases with fidelity 0 and 1, respectively.

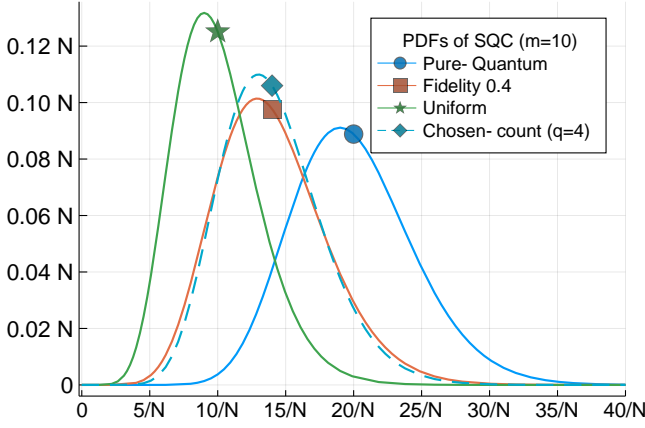


Figure 4: Various PDFs of SQC with $m = 10$

Another adversary of interest, with a quantum computer with fidelity 1, chooses the number q of quantumly-sampled strings, and then uniformly samples the remaining $m - q$ ones. We denote the corresponding random variable SQC as $X_{C,m,q}$ and denote the quotient q/m as the “pseudo-fidelity” of the experiment.

3.1. Statistics of interest

Table 3 shows the expected values and variances of the SQC of m strings, for samplings of type U , Q , F and C . The results are computed in Appendix B. For the first three sampling types the mean and the variance are proportional to m , since each isolated circuit evaluation is assumed independent of the other. It is worth noting that the expected value of $X_{C,m,q}$ is the same as $X_{F,m,q/m}$, but the variance is slightly different.

Figure 4 shows the PDFs of SQCs for the U , Q , $C_{q=4}$ and $F_{\phi=0.4}$ samplings of $m = 10$ strings.

3.2. CDFs of sums of i.i.d. variables

The distinguishability analysis that we are aiming for requires calculation of points in the CDF curves. However, in comparison with the simple formula for the CDF of QC-values, the distributions of SQCs need to account for the new parameter m that can specify an arbitrary number of strings. A common approach to handle the increase in complexity is to apply approximations that simplify the analysis and are provably correct in a limit

Table 3: Statistics of SQCs of m strings

| Sampling type | Random variable | Expected value $E(X)$ | Variance $V(X)$ |
|------------------|-----------------|------------------------|---|
| Uniform | $X_{U,m}$ | m/N | m/N^2 |
| Pure Quantum | $X_{Q,m}$ | $2 \cdot m/N$ | $2 \cdot m/N^2$ |
| Fidelity ϕ | $X_{F,m,\phi}$ | $(1 + \phi) \cdot m/N$ | $(1 + \phi \cdot (2 - \phi)) \cdot m/N^2$ |
| Chosen-count q | $X_{C,m,q}$ | $(m + q)/N$ | $(m + q)/N^2$ |

of increasing the number of summed variables.

Central Limit Theorem (CLT). When sampling a large number of independently and identically distributed (i.i.d.) QC-values, the distribution of their sum $X_{*,m[*]}$ can be approximated by a Normal distribution (11), having a Gaussian shaped PDF with mean $\mu = E[X_{*,m[*]}]$ and standard deviation $\sigma = \sqrt{V[X_{*,m[*]}]}$.

$$\mathcal{N}(\mu, \sigma) : \frac{1}{\sigma\sqrt{2 \cdot \pi}} e^{-\frac{1}{2} \left(\frac{x-\mu}{\sigma}\right)^2} \quad (11)$$

At a first approximation, SQCs can be analyzed based on the central limit theorem [BISB+18]. However, the approximation can have noticeable inaccuracy if the number of summed variables is small and/or when evaluating probabilities at the tails of the distribution.

Better approximations and exact formulas. Appendix C derives exact formulas for the PDFs and CDFs of the SQCs under sampling experiments of interest. Even for large m , the formulas are amenable for direct computation in the uniform, the quantum and the chosen-count cases. For the general fidelity case we can use an exact formula for not-too-large m , but when m gets larger we use a Gamma-approximation that yields better results than the CLT.

For the upcoming analysis we are specifically interested in the ability to evaluate the CDF and its inverse. The Gamma distribution, with parameters derived in Appendix C.3, has a wide applicability, namely with the following formula being applicable to several scenarios.

$$CDF[X_*] = P(m \cdot (1 + \phi'), N' \cdot x_m), \quad (12)$$

where P denotes the [lower] incomplete gamma regularized function (further details in Appendix C.2).

Particularly, the formula is:

- Correct for the uniform, the quantum and the chosen-count cases, provided $\phi' = \phi$ and $N' = N$
- A better-than-the-CLT approximation for the general fidelity case (F), provided $\phi' = \phi \cdot v$ and $N' = N \cdot v$, where $v = 1 + \phi \cdot (2 - \phi)$. The transformation of variables ensures that the expected value $(1 + \phi) \cdot m/N$ and the variance $(1 + \phi \cdot (2 - \phi)) \cdot m/N^2$ are as in the non-approximated case.

Figures 5 and 6 illustrate, for an example with $\phi = 0.5$, how much better the Gamma approximation is, compared with the CLT (Gaussian) approximation. (The Gaussian curve also extends to negative values.)

For not-too-large m , we can efficiently evaluate the PDF

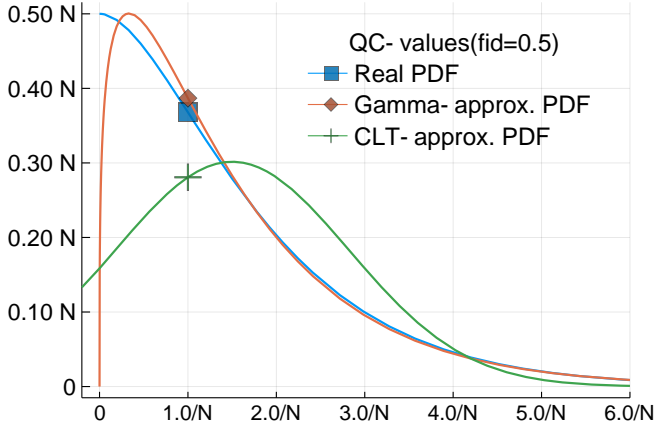


Figure 5: PDF approximations of QC-values $X_{F,.5}$

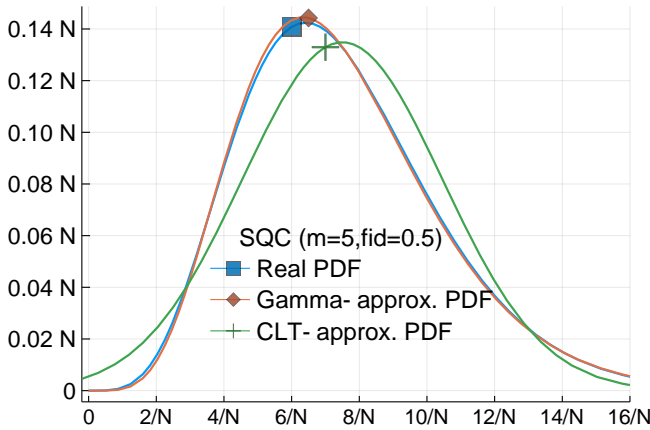


Figure 6: PDF approximations of SQC $X_{F,m=5,.5}$

of the fidelity case (F) as a binomial weighted sum over all possible numbers q of quantumly sampled strings:

$$(1 - \phi)^m \cdot \sum_{q=0}^m \binom{m}{q} \cdot \left(\frac{\phi}{1 - \phi}\right)^q \cdot P(m + q, N \cdot x_m). \quad (13)$$

3.3. Testing honest sampling

Some terminology: We define distinguishability experiments where the baseline question is: *Are we in the presence of an honestly generated set of m strings* [instead of some other malicious or faulty sampling within a well-defined range of behaviors]? In the scope of this experiment/question, we define the following terms: *negative* and *positive* respectively mean rejection and acceptance; *false* and *true* respectively mean incorrect and correct classification. We are particularly interested in the false positive and false negative probabilities:

- **False negative [rate] (FN):** a test rejects an honestly generated sample.
- **False positive [rate] (FP):** a test accepts a sample

generated by a reference malicious or faulty process.

When the **honest** case is characterized by fidelity ϕ , how many (m) strings should be sampled, and what threshold T on the SQCs (X) should be set for an acceptance/rejection test of honest behavior? It depends on:

- what is the reference malicious sampling procedure;
- what FN (ϵ_1) and FP (ϵ_2) rates are set as a goal.

Selecting a reference malicious sampling. As a default reference, we sometimes measure FP with respect to the uniform sampling case (i.e., with fidelity 0). However, the definition of FP should match a higher goal of distinguishability. For example, if intending to maximize the entropy estimate (see Section 5), then a more useful FP will refer to a malicious case that ensures a certain non-zero minimum of entropy. A useful reference of the malicious sampling is $X_{C,m,m,\phi/k}$, where the adversary samples with **pseudo-fidelity** ϕ/k for some $k > 1$, where ϕ is the claimed honest fidelity.

Selecting concrete FN and FP rates. For cryptography applications, the value 2^{-40} is a common benchmark related to statistical security in the “one-shot” security scenario — what can the adversary do if having luck up to an ϵ -likely event? When application goals do not indicate otherwise, we recommend $\epsilon_1 = \epsilon_2 = 2^{-40}$ as a minimum goal for both FN and FP. In specific applications it might be reasonable to allow less stringent security parameters, if explicitly justified.

3.4. Threshold vs. probability

Figure 7 shows, for several fidelity values ϕ , the inverse CDF of the SQC $X_{F,m=10^6,\phi}$ across $m = 10^6$ sampled strings. Each of these curves represents SQC as a function of the FN rate, i.e., of the accumulated probability (horizontal axis) that the random variable $X_{F,m,\phi}$ is at most equal to such threshold value (vertical axis). Figure 8 shows the same for sums across $m = 10^7$ strings. For the uniform case ($\phi=0$) the curve is evaluated directly from its exact formula. For the other cases the curves are obtained from the Gamma approximation. For such high m , the curves look visually the same as if they were obtained from the CLT approximation.

The analysis of the curves illustrates the gap in SQC as the fidelity increases, across the fidelity values in $\{0, .001, .002, .005\}$. Also, comparing the two plots, it is easy to notice the increase in the gap with the increase in the number m of sampled strings, becoming easier to distinguish between two distinct fidelities.

3.5. Sample sizes vs. thresholds

We can now find, for each intended upper-bound ϵ_1 on

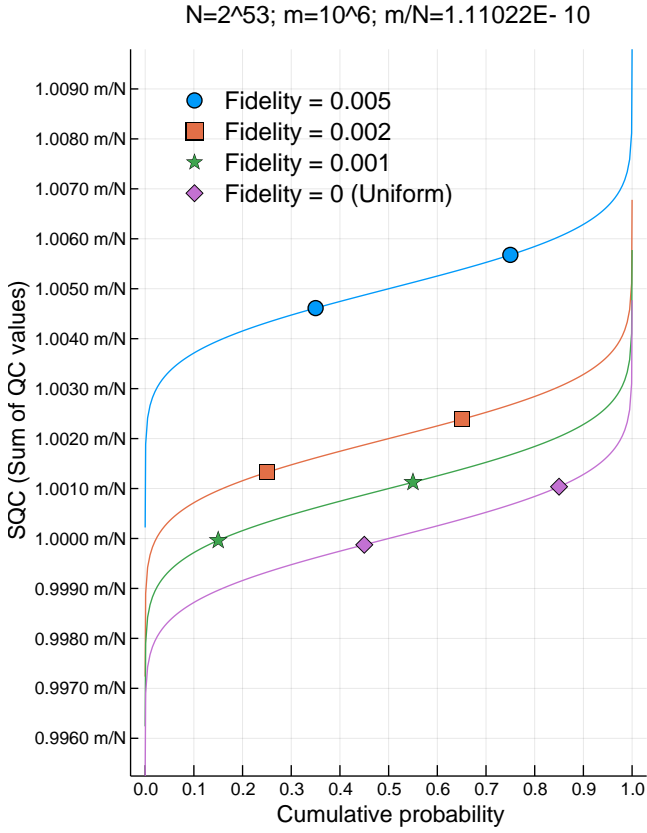


Figure 7: Inverse CDFs of SQC with $m = 10^6$

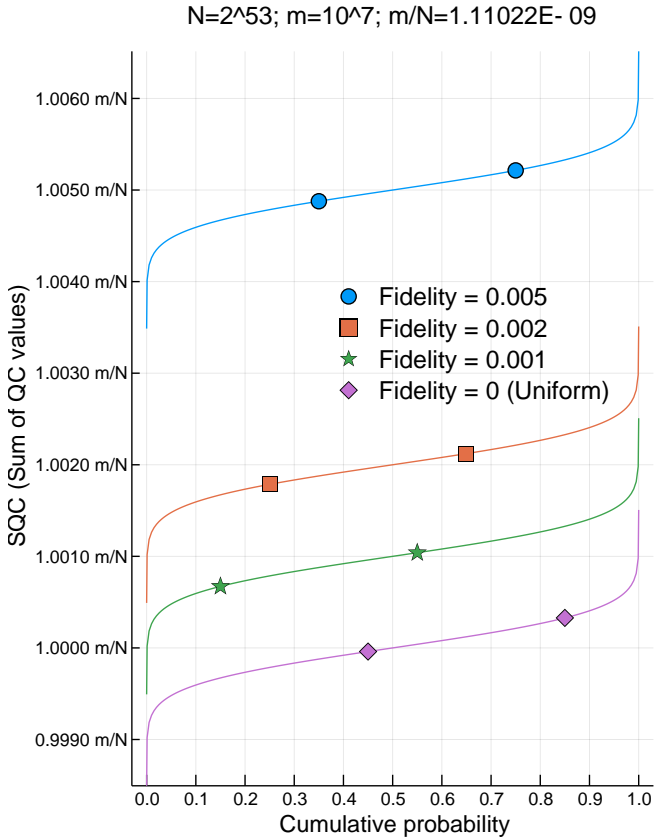


Figure 8: Inverse CDFs of SQC with $m = 10^7$

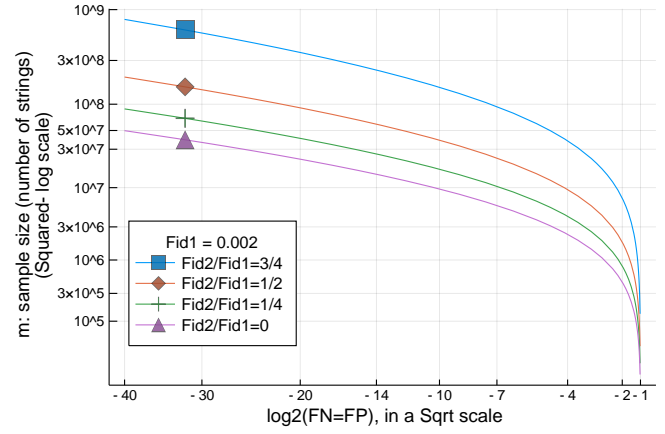


Figure 9: Sample size vs. FN=FP, with $\phi_1 = 0.002$

the FN rate what is the minimal number m of strings needed for an intended FP ϵ_2 , i.e., that a defined malicious execution with lower pseudo-fidelity ϕ_2 is accepted.

Example of FN vs. FP rates. Consider an honest sampling of $m = 10^6$ strings using fidelity $\phi = 0.002$. If the intended FN rate is $\epsilon_1 = 20\%$, then the threshold $T_{F,m,\phi}$ is set at $t = 1.001156 \cdot m/N$, i.e., the value t satisfying $\text{Prob}(X_{F,m,\phi} < t) = 0.2$. If the malicious reference is the uniform case, then the false-positive rate is $\text{FP} = 1 - \text{Prob}(X_{U,m} > t) \approx 12.4\%$. If the malicious reference is the case of fidelity equal to half of the honest fidelity, then we get $\text{FP} = 1 - \text{Prob}(X_{F,m,\phi/2} > t) \approx 43.7\%$.

Figures 9 and 10, respectively for honest fidelities $\phi = 0.002$ $\phi = 0.01$, plot curves of the sample size (number m of sampled strings) vs. the FN=FP rates (ϵ), for a reference honest case X_{F,m,ϕ_1} , and a malicious chosen-count sampling X_{C,m,m,ϕ_2} with pseudo-fidelity ϕ_2 , for $\phi_2/\phi_1 \in \{0, 1/4, 2/4, 3/2\}$. The scales of the axes of the plots are adjusted for a consolidated view of all plotted pseudo-fidelities, while FN=FP range within $[2^{-40}, 2^{-1}]$. Each curve labeled with honest fidelity ϕ_1 and malicious pseudo-fidelity ϕ_2 shows m as a function of $\epsilon = \text{FN} = \text{FP}$, such that there exists a distinguishability threshold t for which $\text{Prob}(X_{F,m,\phi_1} < t) \approx \text{Prob}(X_{C,m,m,\phi_2} > t) \approx \epsilon$.

For simplicity of computation, we applied here the CLT-approximation, which provides a simple formula for m based on the inverse-erf function ((51), (50)), as derived in Appendix C.1. It is worth noting that the analytic results are independent of the string space size N (here assumed to be $N = 2^{53}$), if we assume a sampling with replacement.

Table 4 shows selected examples of sample-size (m) vs. honest-fidelity (ϕ_1) and pseudo-fidelity (ϕ_2). This is a sub-table of the more-detailed Table 14 in Appendix E.

For example, the table shows that about 50 million strings

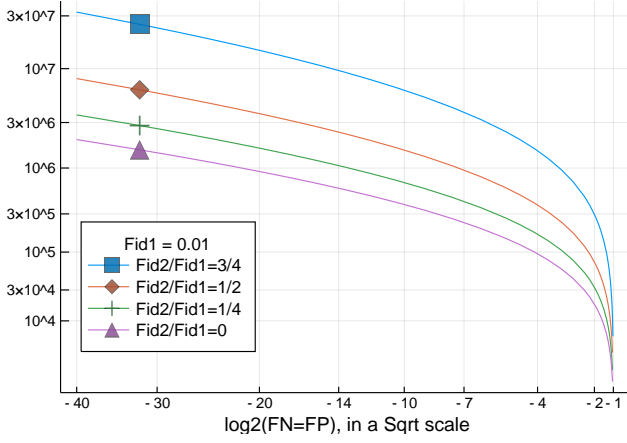


Figure 10: Sample size vs. FN=FP, with $\phi_1 = 0.01$

Table 4: Number of strings for SQC distinguishability (Selected entries from Table 14)

| ϕ_1 | ϵ | Number m of strings to sample | | |
|----------|------------|---------------------------------|--|------------------------------------|
| | | when $\phi_2 = 0$ | when $\frac{\phi_2}{\phi_1} = 10^{-1}$ | when $\frac{\phi_2}{\phi_1} = 1/2$ |
| 0.002 | 2^{-40} | 4.977E+7 | 6.146E+7 | 1.993E+8 |
| | 2^{-30} | 2.273E+7 | 2.807E+7 | 9.102E+7 |
| | 2^{-20} | 9.569E+6 | 1.182E+7 | 3.831E+7 |
| | 10^{-3} | 1.646E+6 | 2.032E+6 | 6.589E+6 |
| 0.01 | 2^{-40} | 2.007E+6 | 2.480E+6 | 8.066E+6 |
| | 2^{-30} | 9.165E+5 | 1.133E+6 | 3.684E+6 |
| | 2^{-20} | 3.858E+5 | 4.767E+5 | 1.551E+6 |
| | 10^{-3} | 6.635E+4 | 8.199E+4 | 2.667E+5 |

need to be sampled in order to obtain FN=FP rates of 2^{-40} , if using a quantum computer with fidelity .002 and if the FP refers to a malicious uniform sampling. The number increases about 4-fold if, instead, measuring FPs with respect to a malicious sampling with pseudo-fidelity of about one half of the honest fidelity. The needed number of QC-values reduces about 25-fold if using instead a quantum computer with fidelity 0.01.

Why considering pseudo-fidelities? Distinguishing the honest vs. the malicious uniform case (fidelity 0) can be useful to ascertain that some of strings resulted from a quantum circuit evaluation. However, for the goal of estimating entropy (see Section 5), it is more relevant to distinguish the honest case from a malicious sampling with some positive pseudo-fidelity $\phi_2 : 0 < \phi_2 < \phi_1$.

4. Low-budget clients

One drawback of the distinguishability experiment considered in the previous section is that it requires computing many QC-values. In this section we consider a different statistic that allows an interesting tradeoff: a *powerful* server computes more QC-values; a *weak* verifying client verifies fewer QC-values. The client verification can be a recomputation of QC-values followed by equality check, but other verifications are conceivable.

In the next subsections we consider one approach for enabling a computationally cheaper verification by clients. Other strategies exist and it would be interesting to investigate which ones may yield useful tradeoffs.

4.1. Truncated QC-values

Suppose the client wants to save work by verifying only (the sum of) a fraction ν of the QC-values. This may be particularly useful in the setting of one string per circuit, as there the cost of computing QC-values is maximal per string. One approach is to set a threshold on the QC-value, and then check only (the sum of) those with higher value.

Truncation thresholds. We denote as truncated QC-value (TQC), with respect to some truncation threshold t , the measure that is equal to the QC-value when it is not-smaller than t , and is 0 otherwise. For each intended verification proportion ν , it is useful to know the matching truncation threshold t . Table 5 shows, for several fidelities, the “truncation threshold” that correspond to each of several verification proportions $\nu \in \{2^{-1}, 2^{-2}, 10^{-1}, 10^{-2}, 10^{-3}\}$. For notation simplicity we let $\tau \equiv t \cdot N$.

Table 5: TQC truncation thresholds

The indicated thresholds are to be divided by N .

| Thresholds | Verification proportion (ν) | | | | |
|------------|-----------------------------------|----------|-----------|-----------|-----------|
| | 2^{-1} | 2^{-2} | 10^{-1} | 10^{-2} | 10^{-3} |
| 0.0 | 0.693 | 1.386 | 2.303 | 4.605 | 6.908 |
| 0.001 | 0.694 | 1.388 | 2.305 | 4.610 | 6.915 |
| 0.002 | 0.695 | 1.389 | 2.307 | 4.614 | 6.922 |
| 0.005 | 0.697 | 1.393 | 2.314 | 4.628 | 6.942 |
| 0.01 | 0.700 | 1.400 | 2.326 | 4.651 | 6.975 |
| 0.02 | 0.707 | 1.414 | 2.348 | 4.695 | 7.039 |
| 0.05 | 0.729 | 1.457 | 2.417 | 4.821 | 7.216 |
| 0.1 | 0.767 | 1.529 | 2.528 | 5.011 | 7.465 |
| 0.2 | 0.850 | 1.675 | 2.739 | 5.331 | 7.852 |
| 0.5 | 1.146 | 2.105 | 3.272 | 5.990 | 8.573 |
| 1.0 | 1.678 | 2.693 | 3.890 | 6.638 | 9.233 |

Since the threshold is measured with respect to the honest distribution of a single QC-value, we can find an exact solution based on the inverse CDF of $X_{F,\phi}$:

$$t = \text{CDF}^{-1}[X_{F,\phi}](1 - \nu) \quad (14)$$

$$= -\frac{1}{\phi \cdot N} \cdot \left(\phi \cdot W_{-1} \left(\frac{\nu \cdot e^{-1/\phi}}{\phi} \right) + 1 \right), \quad (15)$$

where W_{-1} is the real lower branch of the [Lambert W function](#) [DLMF].

For example, for a proportion verification of $\nu = 1/2$ in a case of claimed fidelity 1, the threshold should be set at $\tau = 1.678$. (It is interesting to notice that even though $E(X_Q)$ is $2/N$, the median occurs at about $1.678/N$.) As another example, for a proportion verification of $\nu = 1/100$, the threshold should be set at about $\tau = 6.638$ if the fidelity is 1, and at $\tau = 4.614$ if the fidelity is 0.002.

Consider the random variable $Y_* = I_t \circ X_*$, equal to X_* if $X_* \geq t$ and equal to 0 otherwise. (I_t is the indicator function, equal to 1 if its input is greater than t ; equal to 0 otherwise.) For the sampling of a single string we consider the cases U , Q and F , i.e., when Y is obtained respectively from **Uniform**, **pure-Quantum** and **Fidelity** samplings. Table 6 shows the formula for their expected values and variances. We use $\tau = t \cdot N$. Notice how replacing τ by 0 leads to the formulas in Table 1.

Table 6: Statistics of Truncated QCs

| Statistic | Formula |
|-----------------|--|
| $E[Y_U]$ | $\frac{1}{N} \cdot e^{-\tau} \cdot (\tau + 1)$ |
| $V[Y_U]$ | $\frac{1}{N^2} \cdot (e^{-\tau} \cdot (\tau^2 + 2\tau + 2) - e^{-2\tau} \cdot (\tau + 1)^2)$ |
| $E[Y_Q]$ | $\frac{1}{N} \cdot e^{-\tau} \cdot (\tau^2 + 2\tau + 2)$ |
| $V[Y_Q]$ | $\frac{1}{N^2} \cdot (e^{-\tau} \cdot (\tau^3 + 3\tau^2 + 6\tau + 6) - e^{-2\tau} \cdot (\tau^2 + 2\tau + 2)^2)$ |
| $E[Y_{F,\phi}]$ | $\frac{1}{N} \cdot e^{-\tau} \cdot (\tau^2 \cdot \phi + (\tau + 1) \cdot (\phi + 1))$ |
| $V[Y_{F,\phi}]$ | $\frac{1}{N^2} \cdot e^{-\tau} \cdot (\tau^3 \cdot \phi + \tau^2 \cdot (2\phi + 1) + (\tau + 1) \cdot (4\phi + 2))$ $-\frac{1}{N^2} \cdot e^{-2\tau} \cdot (\tau^2 \cdot \phi + (\tau + 1) \cdot (\phi + 1))^2$ |

Legend: E (**E**xpected value); F (**F**idelity); Q (**Q** pure **Q**uantum); τ ($= t/N$, where t is the **t**runcation **t**hreshold); U (**U**niform); V (**V**ariance); Y (random variable: truncated QC-value).

4.2. Sum of truncated QC values (STQC)

Similarly to how we considered sums of QC-values, we are now interested in the sum of truncated QC values. We use $Y_{m,*}$ to denote the sum of TQC values (STQC) of m strings obtained in an experiment Y_* . Their expected values and variances are obtained by simply multiplying

by m the statistics of the single string.

We also consider the chosen-count case, where an adversary with a fidelity 1 quantum computer chooses in advance the number q (out) of (m) strings that it evaluates quantumly. The expected value and variance of $Y_{C,m,q}$ are given by the corresponding weighted sums from the **uniform** and the **pure-quantum** cases:

$$E(Y_{C,m,q}) = (m - q) \cdot E(Y_U) + q \cdot E(Y_Q) \quad (16)$$

$$V(Y_{C,m,q}) = (m - q) \cdot V(Y_U) + q \cdot V(Y_Q) \quad (17)$$

It is now interesting to calculate the increase in the number of QC-values that a server needs to compute. Table 7 (with selected entries from Table 15 in Appendix E) shows, for several $\text{FN}_{\phi_1} = \text{FP}_{\phi_2}$ rates ϵ , honest fidelities ϕ_1 , fidelity proportions ϕ_2/ϕ_1 , and verification proportions ν , what is the number $m \cdot \nu$ of positive TQC-values to be verified by the client, when the number m of TQC-values computed by the server is higher by a proportion $1/\nu$.

Table 7: Number of client-verified TQC values
(Selected entries from Table 15)

| ϕ_1 | ϵ | $E(m \cdot \nu)$: # client-verified TQC-values | | | |
|----------|------------|---|------------------------------------|----------------------|------------------------------------|
| | | when $\nu = 10^{-1}$ | | when $\nu = 10^{-2}$ | |
| | | when $\phi_2 = 0$ | when $\frac{\phi_2}{\phi_1} = 1/2$ | when $\phi_2 = 0$ | when $\frac{\phi_2}{\phi_1} = 1/2$ |
| 0.002 | 2^{-40} | 7.29E+6 | 2.92E+7 | 2.24E+6 | 8.99E+6 |
| | 2^{-20} | 3.33E+6 | 1.33E+7 | 1.02E+6 | 4.10E+6 |
| | 10^{-3} | 1.40E+6 | 5.61E+6 | 4.31E+5 | 1.73E+6 |
| | 10^{-1} | 2.41E+5 | 9.65E+5 | 7.41E+4 | 2.97E+5 |
| 0.01 | 2^{-40} | 2.97E+5 | 1.19E+6 | 9.34E+4 | 3.78E+5 |
| | 2^{-20} | 1.36E+5 | 5.45E+5 | 4.27E+4 | 1.73E+5 |
| | 10^{-3} | 5.70E+4 | 2.29E+5 | 1.80E+4 | 7.27E+4 |
| | 10^{-1} | 9.81E+3 | 3.95E+4 | 3.09E+3 | 1.25E+4 |

5. Entropy estimation

The previous section focused on distinguishing an honest sampling with a fixed fidelity ϕ_1 from a malicious sampling with a lower pseudo-fidelity ϕ_2 , including 0. This section focuses directly on the goal of entropy estimation from a set of sampled strings, and on that basis deciding parameters for a distinguishability test, and what adversaries to consider. For simplicity, the discussion hereafter assumes a distinguishability test based on SQCs (as described in Section 3). The use of different statistics could change the estimated parameters.

This section is organized as follows: Section 5.1 gives an informal overview of the analysis; Section 5.2 discusses the client perspective; Section 5.3 defines the *pseudo-*

fidelity adversary, for when the sampling budget of the adversary is not enough to observe collisions. Section 5.4 considers the *collisional adversary*, which takes into account the information obtained from observing collisions. Section 5.5 considers the post-processing of the sample and a possible *hash-biasing* attack.

5.1. Overview

The client. With respect to obtaining certifiable entropy, we consider two possible perspectives of the client. Either: (i) it knows how much entropy it wants, and with which assurance it wants to obtain it (FN and FP), and then defines corresponding parameters for a sampling experiment (e.g., the number m of strings to be sampled) and for an acceptance/rejection test (e.g., an SQC distinguishability threshold T); or (ii) it is given a sample of distinct strings claimed to have been sampled from a verifiably fresh circuit, and then, based on their QC-values and on an intended FP rate, it decides a lower bound for the entropy contained in the sample.

The adversary. For either of the above perspectives, the adversary tries to minimize the entropy contained in the published sample, conditioned on satisfying the admissibility criterion of the client (an SQC threshold) with a probability not smaller than FP. We denote the latter as the *FP goal* (or FP constraint).

The adversary of interest is assumed to have a quantum computer with fidelity 1. It produces a sample of m strings where only a small number q of them are actually obtained from quantum evaluation. The sample generation includes adversarial actions of rejection sampling, reordering and biasing, as ways to further reduce the entropy. The $m - q$ non-quantumly-generated strings are obtained pseudo-randomly (with entropy 0) without dependency on the circuit specification \mathcal{C} .

A black-box model. The adversary is assumed to not be able to take advantage of the knowledge of the circuit specification, apart from being able to obtain outputs from its evaluation. Therefore, we idealize the adversary as having a black-box access to the circuit, being able to request its evaluation a number (β) of times. This substantiates the assumption of not being able to compute or estimate QC-values from the circuit specification alone, nor of simulating an evaluation with a correlated probability distribution. Nonetheless, this still allows the adversary to gather some information about QC-values, depending on the sampling budget β , by considering the multiplicity of each occurring string.

Entropy estimation. For concrete parameters of an experiment, we estimate the number of quantumly-generated strings, possibly from various probability distributions. We consider that the strings may have inter-

dependencies, such as those related to sampling without replacement, rejection sampling, and ordering.

5.2. The client

Timing assumption. The continuing discussion retains the assumption that the adversary cannot compute the QC-values of strings by the time deadline to publish them. This requires the circuit specification to not be known in advance by the adversary. For example, one may base this assumption on requiring that the circuit specification is pseudo-randomly generated from the timely output of a public-randomness beacon, if it is reasonable to assume that the beacon is not maliciously colluding with the adversary.

Two perspectives. Using a statistic such as the SQC or STQC described in the previous sections, we consider how to parametrize an experiment and perform an estimation of entropy. For simplicity, hereafter we focus on the SQC statistic. We consider two perspectives:

1. **Decidability problem.** Given a goal of obtaining at least H bits of entropy, with a FP rate of at most ϵ_2 , and accepting a FN rate of up to ϵ_1 for an honest quantum circuit operator with fidelity ϕ_1 , **the client decides the number m of strings** to request from the operator to publish as a sample, and which SQC distinguishability threshold T to use **in order to accept or reject the sample**. The client determines m and T assuming that an adversarial operator (with quantum fidelity 1, for a conservative estimate) will craft a sample with minimum entropy subject to having a probability at least FP of having an SQC greater than the threshold T . The estimate depends on the sampling budget β assumed to be available to the adversary.
2. **Estimation problem.** Entropy can only be estimated retrospectively. Given a circuit specification \mathcal{C} and a published sequence of m strings, the client starts by computing its SQC T . The individual QC-values are either computed by the client or received from an external trusted party that would have computed them. The client also defines, based on the time Δ assumed to have been available to the adversary since learning the circuit specification \mathcal{C} , what is the assumed sampling budget β . The client has its own requirements of FN and FP, and assumes that the adversary may have performed a targeted attack based on those parameters. Thus, the client assumes the final sample includes only a “small” number of quantumly generated strings, minimizing the overall entropy while still attempting to ensure a probability of at least FP of having an SQC at least T . **The client estimates this entropy H .**

Interesting adversaries. We consider that the client is only interested in adversaries that publish samples whose probability of being accepted by the client is at least FP (ϵ_2). Any adversary playing outside this constraint is ignored, since within the intended level of assurance the client will reject the sample. In particular, we ignore adversaries that would publish only a sequence of pseudo-random strings (with overall entropy 0) that would lead to acceptance with probability less than FP.

Number of quantumly-sampled strings. A key step of the analysis is determining the number q of strings that an optimal adversary has included (or will include) in the final sample, and the corresponding expected QC-values and entropies of those strings. The client also assumes that the adversary chooses those strings when knowing the goal/parameters set by the client. In both perspectives, the estimates/parametrizations by the client are based on the assumption that, before the deadline to publish a sample, the adversary cannot compute anything about the QC-values of concrete strings, apart from probabilistic information based on the number of times each string has appeared when quantumly sampling a large (yet feasible) number β of strings from the circuit. The collisional adversary in Section 5.4 considers indeed such information when selecting which quantumly-generated strings to include in the final sample.

5.3. The pseudo-fidelity adversary

We focus first on a specific attack performed by what we denote as the “pseudo-fidelity” adversary. This is tailored to the case where the sampling budget of the adversary is not large enough to enable finding many collisions (repetition of strings) before having to publish a sample. In this subsection we consider the [decidability perspective](#), where the client decides the size m (number of strings) of the sample to be published.

5.3.1. Algorithm

The pseudo-fidelity adversary (\mathcal{A}) operates as follows:

1. **Input.** \mathcal{A} receives four input parameters:
 - (a) β : (budget) \neq quantum evaluations of \mathcal{C} ;
 - (b) m : \neq strings to publish in a **sample**;
 - (c) T : SQC threshold of acceptance by the client;
 - (d) ϵ : FP (maximum false-positive) rate.
2. **Quantum over-sampling.** \mathcal{A} quantumly evaluates the circuit \mathcal{C} , with fidelity 1, in a black-box manner, β times. We call *pre-sample* to the set of obtained distinct strings, which is expected to have approximately $M'_1 \approx N - N/(1 + \beta/N)$ strings. See auxiliary formulas in Section D.2.

3. **Number of quantum strings.** As a function of the input parameters (m, T, ϵ), the adversary \mathcal{A} calculates the minimum number q of quantumly-generated strings (besides the $m - q$ strings to be pseudo-randomly generated) to include in the final sample, such that the client accepts with probability at least ϵ (the FP rate). Recalling, from Section 3, the notation for the SQC random variable X when doing chosen-count sampling, the condition is:

$$q = \min\{q' : \text{Prob}(X_{C,m,q'} \geq T) \geq \epsilon\}. \quad (18)$$

The simplifying assumption here is that the QC-values of these q strings are i.i.d., with expected value $2/N$ and variance $2/N^2$, whereas in fact the budget β and the observation of collisions would enable inferring more detailed information.

4. **Rejection sampling.** Using a secret key known a priori, \mathcal{A} seeds a [cryptographic] pseudo-random permutation (PRP), thus defining a bijective mapping from the set of n -bit strings onto itself. To reduce the expected entropy in the final sample, \mathcal{A} performs “rejection sampling” as follows: (i) \mathcal{A} computes the PRP-output of every string in the pre-sample; (ii) \mathcal{A} orders the M'_1 distinct strings, ascendingly, with respect to their PRP-output; (iii) \mathcal{A} selects, in the corresponding PRP-lexicographical ordering, only the first q strings.
 5. **Positioning of strings.** \mathcal{A} initializes a sample vector \vec{S} of length m (the sample size requested by the client) and pseudo-randomly selects q locations therein. \mathcal{A} then places there the q quantumly-obtained strings selected in the previous step, in the respective devised order. Then, \mathcal{A} pseudo-randomly generates $m - q$ additional strings, distinct from the q already quantumly-obtained strings. \mathcal{A} positions them in the free $m - q$ free positions of the sample. This step adds no additional entropy.
- Note: Section 5.5 mentions a possible additional step of hash-biasing.
6. **Output.** \mathcal{A} outputs the sequence of m strings.

5.3.2. Statistics

We now summarize how the client, with a goal of obtaining a sample with H bits of entropy (e.g., 1024), should parametrize an experiment when assuming the quantum computer operator is a pseudo-fidelity adversary (\mathcal{A}).

Pre-sample size. We assume that \mathcal{A} can sample the quantum circuit at most β times in the allotted time window. For example, if one circuit evaluation counts as one cycle, then sampling $\beta = 2^{26}$ strings within a time window of 60 seconds requires a quantum computer with

a frequency of about 1.12 MHz. For $\beta < \sqrt{N}$, we assume for simplicity that the the number $M'_1 \approx N - N/(1 + \beta/N)$ of obtained distinct strings is β (see Section D.2).

QC-values and pseudo-fidelity. With $\beta \ll N$, the expected QC-value in the pre-sample is very close to what was determined in (9), i.e., $E \approx 2/N$. Since the client wants to accept an honest sample with high-probability (i.e., low FN rate), the corresponding SQC threshold necessarily allows some probability of it also being achieved with a pseudo-fidelity slightly lower than the honest fidelity. When the adversary does rejection sampling to select q strings, then by definition the pseudo-fidelity is $\phi_2 = q/m$. The client assumes the adversary uses the smallest pseudo-fidelity that will still pass the SQC threshold test probability non-lower than the FP rate ϵ . Section D.2 considers a more detailed approximation (124) for E , obtaining $E = (1/N) \cdot (2 + b)/(1 + b)$, where $b = \beta/N$ is the budget factor. However, compared with $2/N$ the correction is only significant when it is already relevant to consider the more sophisticated collisional adversary from Section 5.4, which takes advantage of collisions observed in the pre-sampling stage.

Entropy estimation. The original expected entropy per each of the obtained distinct strings in the pre-sample depends on the sampling budget β . Let h_β denote the expected entropy for a thought experiment where the adversary would now output a single string uniformly selected from the pre-sample. This is $h_\beta \approx n - (1 - \gamma)/\log(2) \approx n - 0.61$ (as determined in (9) in Section 2.3) if $\beta < \sqrt{N}$, or up to $h_\beta \approx n$ when β is so large that the pre-sample contains almost all N strings. However, \mathcal{A} has performed “rejection sampling” to reduce the expected entropy per string (and thus of the final sample).

An initial intuition is that the rejection sampling induces the selected PRP-outputs to start with about $\log_2(M'_1/q)$ zeros (meaning we can discount about $\log_2(M'_1/q)$ bits of entropy per string). Also, the ordered selection reduces the space of possible vectors by a factor $q!$, increasing the probability of each possible one by $q!$. This would lead to approximating the expected entropy as follows:

$$q \cdot (h_\beta - \log_2(M'_1/q)) + \log_2(q!) \quad (19)$$

However, a lower value is obtained if we consider an iterated procedure, one string at a time (i.e., using $q = 1$), repeated the original q times. In the iterated case, at the i -th selection the entropy reduction from the above formula would be $\log_2(M'_1 - i)$, leading to an apparent overall reduction of $\log_2(M'_1!/(M'_1 - q)!)$.

The Shannon entropy of a sampling procedure is the expected value of the negative logarithm (base 2) of the probability of obtained samples. This logarithm can be interpreted as a summation of logarithms, where

each new logarithm applies to the probability of a new string being selected, conditioned on the strings already selected. The logarithm for the probability of the overall sample is itself a random variable, which has not only an expected value (Shannon entropy) but also for example a variance. In practice it may be relevant to measure something like the minimum possible entropy, or a bound for which there is a overwhelming probability of the variable being larger than it. Some details about this variable are considered in Section D.3.

For simplicity, and being conservative in the estimation of Shannon entropy, in the subsequent discussion we focus on the approximation obtained by iterating formula (19) one string at a time. We also pug in a minor correction (see Section D.3.1) to the apriori average entropy h_β per string (i.e., before rejection sampling), due to the reduction of the set from which the i -th string is selected. In summary, we consider the following approximation:

$$H_{\beta,q} = \underbrace{\log_2 \left((2^{h_\beta})^q \right)}_{\approx q \cdot h_\beta} - \log_2 \left((M'_1)^q \right), \quad (20)$$

where h_β is the average entropy per string expected for the pre-sample of distinct strings obtained after β quantum evaluations of the circuit, and where $\log_2(x^q)$ represents the binary logarithm of the descending factorial of x order q , i.e., of $x \cdot (x - 1) \cdot \dots \cdot (x - q + 1)$. A more accurate estimate could be based on simulation, as mentioned in Section D.3. For example, there we show that with fidelity 0 and $q = 1$ both formulas are a non-tight lower-bound of the expected entropy.

Parametrization. Solving for q yields the (approximate) minimum number of quantumly-generated strings that an **interesting adversary** will use. The client assumes that the sample of m strings will be produced by a **chosen-count** method, using exactly $\lceil q \rceil$ quantumly generated strings (i.e., pseudo-fidelity $\phi_2 = m/q$), selected (and ordered) upon rejection sampling from a set of $\approx M'_1$ strings. Thus, the client determines the parameters m , and T , respectively using the approximated equations (51), and (48) or (46), in Section C.1, as in the following examples.

5.3.3. Initial examples

Example 1. Let $n = 53$ and $\beta = 2^{26}$, which implies $h_\beta \approx 52.39$ and $M'_1 \approx 2^{26.00}$. Instantiating this in equation (19) for a goal of $H = 2^{10}$ bits of entropy and solving for q yields $q \approx 38.80$, which rounded up is $\lceil q \rceil = 39$ (the assumed number of quantumly-generated strings that the adversary will include in the final sample).

When $\text{FN} = \text{FP} = \epsilon = 2^{-40}$, the needed sample size m can, by the CLT, be approximated as in (51) in Appendix C.1, using $\phi_2 = \lceil q \rceil / m$. Let ϕ_1 be the quantum

fidelity claimed by the honest operator. Then:

- $m \approx 2.014\text{E}+6$ for $\phi_1 = 0.01$
- $m \approx 4.981\text{E}+7$ for $\phi_1 = 0.002$

The corresponding SQC thresholds can be obtained from (48) in Section C.1, using $\mu_2 = (m + q)/N$ and $\sigma_2 = \sqrt{m + q}/N$. Comparing against Table 8, the values obtained for m are only slightly higher than (but quite close to) those obtained for $\epsilon = 2^{-40}$ and $\phi_2/\phi_1 = 0$. This is expected, since $\lceil q \rceil = 39$ makes the ratio ϕ_2/ϕ_1 very small, namely $1.94\text{E}-3 \approx 39/(0.01 \cdot 2.014\text{E}+6)$ and $3.92\text{E}-4 \approx 39/(0.002 \cdot 4.981\text{E}+7)$, respectively when ϕ_1 is 0.01 and 0.002. Therefore, it is expected that a small factor increase in m will yield a large factor increase in estimated entropy.

Table 8: Number of strings for SQC distinguishability (Selected entries from Table 14)

| ϕ_1 | ϵ | m for $\phi_2 = 0$ | m for $\frac{\phi_2}{\phi_1} = 1/100$ | m for $\frac{\phi_2}{\phi_1} = 1/4$ | m for $\frac{\phi_2}{\phi_1} = 1/2$ |
|----------|------------|----------------------|---|---------------------------------------|---------------------------------------|
| 0.002 | 2^{-40} | 4.98E+7 | 5.08E+7 | 8.85E+7 | 1.99E+8 |
| | 10^{-1} | 1.65E+6 | 1.68E+6 | 2.93E+6 | 6.59E+6 |
| 0.01 | 2^{-40} | 2.01E+6 | 2.05E+6 | 3.57E+6 | 8.05E+6 |
| | 10^{-1} | 6.63E+4 | 6.77E+4 | 1.18E+5 | 2.66E+5 |

Example 2. If the client wants $H = 2^{20}$ bits of entropy, then we get $q = 39\,733.1$. Using the same FN=FP rates $\epsilon = 2^{-40}$, and assuming that \mathcal{A} will use $\lceil q \rceil = 39\,734$ quantumly-generated strings, leads to:

- $m \approx 7.980\text{E}+6$ for $\phi_1 = 0.01$
- $m \approx 8.486\text{E}+7$ for $\phi_1 = 0.002$

We note that the last example for m already exceeds the considered budget $\beta = 2^{26}$, meaning that such (β, m) parametrization would not be suitable for a real experiment. The adversary client should then assume a higher budget to the adversary (e.g., possibly by giving more time for sampling). Consequently, the client should re-estimate m and possibly also start taking into account the advantage that the adversary may obtain from observing collisions in the over-sampling stage.

5.3.4. Other examples

We show now a few examples of deriving the parameter q from Table 8 (which has a few selected entries from Table 14), based on the sample size (total number m of strings) calculated with respect to the SQC statistic, for several pseudo-fidelities ϕ_2 and FN=FP rates.

Example 3. Let $(\phi_1, \epsilon, \phi_2/\phi_1) = (0.002, 10^{-1}, 1/100)$. For FN=FP rates of $\epsilon = 1/10$, one needs $m = 1.68\text{E}+6$

sampled strings to distinguish between the honest sampling $X_{F,m,0.002}$ with fidelity $\phi_1 = 0.002$, and a malicious chosen-count sampling $X_{C,m,m\phi_1/100}$ with pseudo-fidelity ϕ_2 equal to one hundredth of ϕ_1 (i.e., $\phi_2/\phi_1 = 1/100$). Such malicious case contains only $q = 34 (= \lceil m \cdot \phi_1/100 \rceil)$ strings generated from a correct circuit evaluation.

Example 4. Let $(\phi_1, \epsilon, \phi_2/\phi_1) = (0.002, 10^{-1}, 1/4)$. Compared to Example 3, increasing m by a factor of about 78 %, to $m = 2.93\text{E}+6$, provides the same FN=FP rates (0.1) but when FP refers instead to a malicious pseudo-fidelity ϕ_2 equal to 1/4 of the honest fidelity ϕ_1 . That corresponds to 1 464 quantumly generated strings, which is 40 times more than in Example 3.

Example 5. The cost in sample size is about linear in the \log_2 base of the FN=FP rates. Let $(\phi_1, \epsilon, \phi_2/\phi_1) = (0.002, 2^{-40}, 1/100)$. Consider an FP=FN goal of $\epsilon = 2^{-40}$, for cryptographic suitability. More than 49.7 million strings need to be obtained for any amount of entropy to be present in the malicious case, if supposing $\phi_1 = 0.002$. For example, for a pseudo-fidelity $\phi_2 = \phi_1/100$, the total number of strings is about $m = 5.08\text{E}+7$ strings, meaning about 1 016 quantum-sampled strings. Overall the sample size is about 30 times larger than in Example 3, to increase the FP=FN rates from 10^{-1} to 2^{-40} .

Example 6. Let $\phi_1 = 0.01$. A substantial improvement is possible by increasing the honest fidelity ϕ_1 reference. For example, for $\phi_1 = 0.01$ (five times larger than in example 3), the needed total number m of strings is about 2 million (i.e., about 25 times less than) for the same ϕ_2/ϕ_1 ratio as in Example 5. However, to satisfy the FP constraint the malicious adversary would (in this example) also be using a higher pseudo-fidelity, equal to $0.01/100$. One would then assume there are 205 quantum-sampled strings present in the sampled string set.

5.4. The collisional adversary

We consider now how the observation of collisions during over-sampling may give an advantage to the adversary.

Informal description. The adversary \mathcal{A} uses a quantum computer with fidelity 1 to evaluate the circuit \mathcal{C} “many” times (β), until obtaining “many” collisions. The output strings with most collisions have a higher expected QC-value (and lower expected entropy) than those with fewer collisions. Based on this, fewer quantumly generated strings in the final sample can achieve a higher SQC. \mathcal{A} organizes the strings in bins, one for each multiplicity of occurrence (i.e., bin i becomes the set of strings that appeared, each, exactly i times). Depending on the tally of multiplicities, i.e., the vector of numbers of strings across bins, \mathcal{A} chooses from which sets of bins to select strings for the final sample, along with applying rejection sampling.

When a small sampling budget does not lead to collisions, the collisional adversary corresponds to the pseudo-fidelity adversary from Section 5.3.1. Conversely, in a theoretical extreme of a sampling budget being a very large exponential in the number of qubits, each string would get a multiplicity sufficiently apart from the multiplicities of other strings. From those multiplicities the adversary could estimate with high accuracy the QC-values of each string. This would enable a straightforward simulation of sampling from a circuit evaluation, while in fact only making a pseudo-random selection with overall zero entropy.

5.4.1. Algorithm

The collisional adversary operates as follows:

1. **Input.** As in Section 5.3.1, \mathcal{A} receives the **input parameters**: $\beta; m; T; \epsilon$.
2. **Quantum over-sampling.** \mathcal{A} uses its sampling budget to obtain a sequence of β strings, called the *pre-sample*, which may have repetitions (i.e., collisions). \mathcal{A} organizes the strings into bins. Bin c is the set of strings that appeared with multiplicity c in the pre-sample.

Let M_c denote the expected number of strings in bin c . For example, with $n = 53$ qubits we have:

- $\beta = 2^{28} \Rightarrow (M_1, M_2) \approx (\beta - 2^4, 2^3)$
- $\beta = 2^{32} \Rightarrow (M_1, M_2) \approx (\beta - 2^{12}, 2^{11})$
- $\beta = 2^{36} \Rightarrow (M_1, M_2, M_3) \approx (\beta - 2M_1 - 3M_2, 2^{19} - 12, 4)$

We abuse notation and also let M_c denote the actual number of strings obtained with each multiplicity in a given experiment. We have:

$$\beta = \sum_{c=1}^{\beta} c \cdot M_c. \quad (21)$$

We use a prime in superscript (e.g., as in M'_c) to indicate the union of bins of multiplicities larger or equal to a certain value. For more general union of bins we can simply use a set, instead of an integer, as index. For example: $M'_c \equiv M'_{\{c, c+1, \dots, \beta\}}$.

3. **Number of quantum strings.** The adversary decides, from the pre-sample with M_c distinct strings in each bin c , totalling M'_1 distinct strings, how many (q) quantumly-obtained strings, and from which unions of bins, to include in the final sample. An adversarially optimal selection takes into account that the expected QC-value and entropy of quantumly sampled strings also varies across the bins. Tendentiously the strings with higher multiplicity are preferable in terms of having higher QC value and lower entropy. However, an optimal decision can be more intricate considering the rejection

sampling **step ahead**, which depends on the number M_c of strings available in each bin c , and on the SQC threshold required by the client to accept a sample.

Concretely, as a function of the input parameters (m, T, ϵ) , and of the tally (M_1, M_2, \dots) of collisions in the pre-sample, \mathcal{A} will determine a sequence $\langle u_1, u_2, \dots \rangle$ of subsets (possibly only one) of multiplicities, from whose corresponding unions-of-bins to pseudo-randomly sample, and determine how many strings to select from each such union. In other words, \mathcal{A} needs to determine two (jointly optimal) non-empty same-length sequences:

- $\vec{u} \equiv \langle u_1, u_2, \dots \rangle$ (or simply $\langle u_1 \rangle$), where each u_i is a subset of possible multiplicities $\{1, \dots, \beta\}$.
- $\vec{q} \equiv \langle q_{u_1}, q_{u_2}, \dots \rangle$, where each q_{u_j} is a positive integer, such that $q \equiv \sum_{j=1, \dots, |\vec{u}|} q_{u_j}$ is the number of quantumly-obtained strings to be included in the final sample (not counting strings that, although obtained in the over-sampling **step** but not selected in the rejection sampling **step**, may in the subsequent **step** be, by coincidence, pseudo-randomly selected.)

Note on various options: The pseudo-fidelity adversary described in Section 5.3.1 uses $\vec{u} = \langle \{1, \dots, \beta\} \rangle$, i.e., all q strings are selected from within the set of all pre-sampled distinct strings (regardless of multiplicity); conversely, the collisional adversary is allowed a more intricate choice across different unions of bins. We define that the *optimal* collisional adversary is one that makes an optimal choice of the vectors \vec{u} and \vec{q} , for the purpose of minimizing entropy while satisfying the FP goal.

4. **Rejection sampling.** From each union set u_j , the adversary uses a differently seeded pseudo-random number generator to obtain q_{u_j} strings by rejection sampling. Specifically, \mathcal{A} selects the lexicographically first q_{u_j} strings upon application of the pseudo-random permutation. See Section D.3.2 for a discussion of other options.
5. **Positioning of strings.** \mathcal{A} initializes a sample vector \vec{S} of length m (the sample size requested by the client) and pseudo-randomly selects a vector of $q \equiv |\vec{q}|$ distinct positioned therein. \mathcal{A} then sequentially places in those positions the q quantumly obtained strings selected in the previous step, which possibly came from various bins, in the respective devised order (namely, considering the lexicographic ordering respective to the pseudo-random permutation used for rejection sampling in each union of bins). Then, \mathcal{A} pseudo-randomly selects $m - q$ other strings, distinct from the already selected q strings, to complete a final sample with overall m strings.
6. **Output.** \mathcal{A} outputs the sequence of m strings.

5.4.2. Statistics per bins or unions of bins

Size of bins. Appendix D.2 shows experimentally obtained formulas for M_c , as functions of the string space size N and the sampling budget factor $b = \beta/N$. One case of interest is the union of bins whose multiplicity is *at least* a certain value (c). In those cases we use a prime in superscript (e.g., as in M'_c) to indicate that the statistics refer to said union of bins. For example, the expected number M'_c of distinct strings that appear with multiplicity at least c is approximately equal to:

$$M'_c = \sum_{i:i \geq c} M_i. \quad (22)$$

QC-values. Appendix D.2 shows experimentally obtained formulas for the expected QC-value (E_c) and variance (V_c), for each bin c . When the budget β is significantly smaller than the string space, the expected QC-value and the variance remain very close, respectively, to $(c+1)/N$ and $c \cdot (c+1)/N^2$, within each bin c . However, for larger sampling budgets those values start to noticeably differ. It is also relevant to consider the special case of the expected average QC-value A'_c in the union of bins with at least a certain multiplicity c . This is approximately equal to:

$$A'_c \equiv \left(\sum_{i:i \geq c} A_i \cdot M_i \right) / M'_c. \quad (23)$$

Entropy. The expected entropy H_c per string decreases with the multiplicity. However, the entropy of each string in the pre-sample is indistinguishable across the strings within the same bin, i.e., before the deadline to publish a sample. Nonetheless, the adversary will still affect the probability distribution with which the strings will appear in the final sample, by using rejection sampling. Technically, the entropy of a string as measured in the pre-sample is not the same as measured in the final sample. Particularly, depending on the rejection sampling technique, the entropy in the final sample will also depend on the number q_u of strings to select from each union u of bins.

By definition, an *optimal* collisional adversary is the one that optimally selects the sequence of unions of bins in a way that minimizes the overall expected entropy in the final sample, while subject to the FP goal. For the purpose of these notes we find sufficient to highlight the effect of a simple collisional choice — $\vec{u} = \langle \{2, \dots, \beta\} \rangle$ (i.e., selecting strings from those that appeared at least twice in the pre-sample) — that already outperforms the pseudo-fidelity attack when the sampling budget β is sufficiently large, such as 2^{32} when considering $n = 53$ qubits.

Using a logic similar to the one used for the analysis of the pseudo-fidelity adversary, we can consider a first

approximation of the entropy contributed by the first ordered sequence of q_{u_1} strings selected from the first union of bins as being about:

$$q_{u_j} \cdot (h_{u_j} - \log_2(M_{u_j}) + \log_2(q_{u_j})) - \log_2(q_{u_j}!) \quad (24)$$

where h_{u_1} is the expected a priori average entropy per string in the union of bins in u_1 .

For a more conservative and still simple estimate we can consider the result of iteratively applying the previous formula, thus getting:

$$\underbrace{\log_2 \left(\left(2^{h_{u_1}} \right)^{q_{u_1}} \right)}_{\approx q_{u_1} \cdot h_{u_1}} - \log_2 \left(\left(M_{u_1} \right)^{q_{u_1}} \right), \quad (25)$$

where $x^{\underline{q}}$ is the descending factorial of x order q . The expressions are similar in look to formulas (19) and (20), which considered $\vec{u} = \langle u_1 \rangle = \langle \{1, 2, \dots\} \rangle$, but now we consider separate bins.

The corrections discussed in Section D.3 also apply. The entropy contributed by strings selected across several unions of bins also depends on the previous selections across other unions of bins. Concretely, the initial entropy h_{u_j} considered for the first string in a union u_j on bins depends on the number of strings already selected, and from which unions of bins.

Asymptotically large budget. It is instructive to consider the asymptotic limit of large sampling budgets, i.e., when β is a large enough exponential in the number of qubits. Each string s will tend to appear in an individual bin with multiplicity c_s approximately equal to $\beta \cdot p_s$, where $p_s \equiv \text{Prob}(s \leftarrow C)$ is the QC-value of string s with respect to the circuit \mathcal{C} . Thus, the adversary can estimate that the QC-value of each string is approximately $\hat{p}_s \approx \beta/c_s$.

In fact, the asymptotically large β would even allow an exact simulation of the circuit evaluation, as follows.

1. Pseudo-randomly simulate a binomial number q of strings to obtain from quantum evaluation. The binomial has parameters m and ϕ , to simulate how many strings, out of m , would be from correct quantum evaluation in an experiment with fidelity ϕ .
2. Pseudo-randomly simulate q uniform floating point numbers between 0 and 1, as points in the inverse-CDF of the QC-values, and determine what are the correspondingly selected strings.
3. Output a sequence of m strings, composed of a pseudo-random positioning of the initial q strings, and then interleaved by other $m-q$ pseudo-randomly selected strings simulating a uniform selection of distinct strings.

The above described sample would be cryptographically indistinguishable from a honest sample, implying that not only it would pass an SQC test with the same FP rate as the FN rate set for the honest case, as well as it would do so for any other practical statistical test.

5.4.3. Comparison of adversaries

The pseudo-fidelity adversary is a special case of the collisional adversary, when using $\vec{s} = \langle \{1, \dots, \beta\} \rangle$ and then selecting q is as the minimum possible. We conjecture that in the black-box evaluation model this is optimal for a sampling budget of the order $\beta \ll \sqrt{N}$ since there is no information gained from collisions. Obtaining a few collisions is possible in practice by evaluating the circuit a number of times of at least the order of the square-root of the string space. However, for $n = 53$ qubits, where a distinguishability from uniform with FP rate $\epsilon = 2^{-40}$ and fidelity $\phi_1 = 0.002$ already requires sampling approximately $2^{25.6}$ times, it is conceivable that an adversary would in fact be able to sample more strings, say, up to 2^{32} , within the allowed time for sampling.

The collisional adversary takes advantage of observed collisions. Below, we show some examples comparing the pseudo-fidelity adversary vs. a simple collisional adversary that simply selects strings from those that have collided twice, i.e., using $\vec{s} = \langle \{2\} \rangle$. (For higher budgets an optimal collisional adversary may be more successful by using a variety of bins and their unions.)

Pseudo-fidelity vs. collisional. Consider parameters (m, T, FP) such that a pseudo-fidelity adversary (P) is compelled to include q_1^P (e.g., 1024) quantumly-generated strings, and then pseudo-randomly obtain the remaining $q_0^P = m - q_1^P$ strings.

We now ask: how many quantumly-generated strings a collisional adversary would actually include in the sample if its budget induces a large enough number of collisions? The answer depends on several quantities. For the given budget (implicit, not shown in the indices), let:

- A_c be the expected QC-value of strings in bin c , the values of which are determined in Section D.2.1;
- q_c^C be the number of strings that the collisional adversary will select from bin c ;
- $A_{\text{PRG}} \approx 1/N$ be as A_c , and q_{PRG}^C be as q_c^C , but with respect to the set of all strings that the adversary will not select from bins with positive multiplicity, and of all strings not output by quantum evaluation. This is the set from which the adversary will select strings directly by pseudo-random generation.

Consider a simplified collisional adversary that will, for

the same final sample size m , use Δ fewer quantumly generated strings, all from bin c . Then we have:

$$q_c^C = q_1^P - \Delta \quad (26)$$

$$\text{SQC}^P = q_{\text{PRG}}^P \cdot A_{\text{PRG}} + q_1^P \cdot A_1 \quad (27)$$

$$\text{SQC}^C = (q_{\text{PRG}}^P + \Delta) \cdot A_{\text{PRG}} + q_c^C \cdot A_c, \quad (28)$$

where SQC^A denotes the expected sum of QC-values, and \mathcal{A} refers to either the pseudo-fidelity (P) or the collisional (C) adversary. The above system yields:

$$q_c^C = \frac{N \cdot A_1 - 1}{N \cdot A_c - 1} \cdot q_1^P. \quad (29)$$

Example 7. Consider a setting with $n = 53$ qubits. Table 9 shows, for two different budgets (2^{32} and 2^{36}), the estimated entropy for collisional attacks of various degrees (i.e., various multiplicities c they take advantage of). The calculation is for a case with $q_1^P = 1024$ (i.e., when the SQC threshold requires a contribution of $2 \cdot 1024/N$ from strings quantumly-generated by a pseudo-fidelity adversary).

The number M_c of strings obtained with multiplicity c depends on the budget β . For example, evaluating the circuit 2^{32} times yields about 2^{11} collisions. The expected entropy h_c per string also varies with the budget and the multiplicity c , as determined in Section D.2.2. In the table, the precision shown for M and $N \cdot A$ was tailored in each case to highlight how small is the correction compared to the approximations $M_1 \approx \beta$ and $N \cdot A_c \approx 1 + c$.

Some observations:

1. For both budgets the expected QC-values of strings are still very close to $(c + 1)/N$, but they would become noticeably different for high enough budget.
2. For $q_1^P = 1024$, the attack with $c = 2$ is not possible if $\beta < 2^{31}$, since then $q_2^C = 512$ would be greater than the expected number of collisions.
3. The higher the budget the lower the entropy.

Table 9: Comparison pseudo-fidelity vs. collisional

| Assuming $q_1^P = 1024$ | | | | | | |
|-------------------------|-----|------------------|---------------|--------|------------------|---------------------|
| β | c | M_c | $N \cdot A_c$ | q_c | h_c | H_c |
| 2^{32} | 1 | $2^{31.9999999}$ | 1.999999 | 1024.0 | ≈ 52.390 | $\approx 20\ 879.4$ |
| | 2 | $2^{10.9999999}$ | 2.999999 | 512.0 | ≈ 51.337 | $\approx 20\ 753.5$ |
| 2^{36} | 1 | $2^{35.99998}$ | 1.99998 | 1024.0 | ≈ 52.390 | $\approx 16\ 783.4$ |
| | 2 | $2^{18.99997}$ | 2.99998 | 512.0 | ≈ 51.337 | $\approx 16\ 556.9$ |

The approximation H_c is based on (24), i.e., the iterated application of (25) in each bin. The direct application (24), one string at a time, would yield a higher estimate, by a factor of up to about 10 % in each case.

A more relaxed estimation. We have considered an adversary with a quantum computer with fidelity $\phi' = 1$. But a client may want to assume that the adversary can only quantumly evaluate with a lower fidelity, e.g., $\phi' = 0.5$. Then, in a model where any faulty evaluation yields a uniformly random string, the previous formulas (9) for expected entropy h per string output by the quantum computer would have to be adjusted. Then, to achieve the entropy reduction the adversary would need to have higher sampling budget.

5.5. Final randomness for applications

What use may a client make of a list of millions of strings that may potentially include several hundreds or thousands of bits of fresh entropy? We do not explore in these notes the interesting use of information theoretic randomness extractors. However, from a practical standpoint and considering cryptographic use, we recommend the use of a cryptographic combination of the entropy, into a bit-string with approximately full entropy. For example, this can be a 512-bit string with about 511 or 512 bits of entropy. (We propose to assume 511.37 bits, as expected for a random function with 512 bits of output.) For practical purposes, this is enough as a seed of a pseudo-random number generator that can then produce a much larger string indistinguishable from random (by whoever does not know the seed). A combination performed by direct application of an efficient cryptographic hash function is a candidate with merit but susceptible to a bias attack.

Hash bias attack. If the adversary \mathcal{A} anticipates that the client will extract entropy from the sample by applying a fast-to-evaluate compressing function without secrets, then it can induce a further reduction in entropy. For example, consider that the client uses a cryptographic hash function, such as the Secure Hash Algorithm 3 (SHA-3) version with 512 bits of output, to hash the sample and use it as the actual randomness output by the protocol. In that case, \mathcal{A} can try many modifications of the one or two of the last pseudo-random strings in order to bias the hash of the sample, e.g., making it satisfy a secret predicate with small positive probability (e.g., of about 2^{-64} if it can still perform 2^{64} hash computations within the deadline). This makes the hash output be from a set of reduced size, e.g., about $2^{511.37-64}$ instead of $2^{511.37}$. For example, the adversary could induce the first 64 bits of the hash to be a certain secret known only to the adversary. This would reduce the entropy of the output by about 64 bits. In practice this is not problematic for applications that intend to use a seed not required to have more than, say, 400+ bits of entropy.

Nonetheless, we describe two possible mitigations:

- If the application allows the client to wait for the calculation of QC-values, then the client can include

(at least a few of) the QC-values of the strings as part of the hash input. This can be impractical for some applications, due to the required delay in computing the QC-values.

- Using a verifiable delay function for the hash, the adversary does not have enough time to compute it before it has to publish the sample of strings.

Alternative post-processing. Alternatively to hashing, the client can instead include a secret key (if one exists) as part of the hash input, to prevent the operator from limiting the size of the image space. A standard method for this approach is to use a hash-based message-authentication code. But if it is a one-time use secret, it is enough to prepend it to the rest of the sample before hashing. Actually, the entropy is not lost if the client reveals the secret at this stage, as long as it was unpredictable by the adversary before it had to publish the sample of strings. In an application setting where it is preferable to also prevent the client from biasing the output, then the secret can be committed in advance, before the circuit is sent to the quantum computer operator.

5.6. Classes of adversaries

Let an adversary be called “optimal” within a class if, while satisfying the FP constraint, it minimizes the entropy of the sample.

Security reduction. These notes do not provide a complexity-theoretic reduction. Such a reduction would have to rule out the existence of adversaries much stronger than those we consider here. Instead, we make the entropy estimation for a concrete efficient adversary, arguably optimal within an interesting class. We leave as an open problem investigating how large this class is. A reduction by Aaronson [Aar19] guarantees a minimum of a few bits of entropy, in the setting of one string per circuit. We consider, instead, repeated sampling from the same circuit, as described in Section 2.4.

Class “A” of adversaries. We define class “A” as the class of adversaries (parametrized by a sampling budget β) for which the *optimal* adversary is a collisional one. We hypothesize that this class captures the range of adversaries of practical concern. We also hypothesize that class A includes the set of efficient adversaries that only access the quantum computer via a black-box interface. While we do not know of any efficient adversary, outside class A, that is better than the collisional adversary, we do not rule out that possibility. We hypothesize that any efficient adversary outside of this class would need to use a non-trivial (currently unknown) mathematical trick taking advantage of the circuit specification \mathcal{C} . The intuition for this hypothesis is conveyed below by an analogy.

An analogy. We do not prove how general or restrictive the class \mathcal{A} is with respect to affecting the distribution of QC-values, nor do we attempt to relate it to a complexity theoretic argument. However, we provide an intuitive argument by making an analogy with the properties of Carter-Wegman *universal hashing* [CW77].

- **Universal hashing:**

1. **Member of a large class.** The hash function is uniformly selected from a large family of hash functions, all with the same output range.
2. **Equal distribution of output values.** For each hash function, each possible output has the same number of pre-images.
3. **Advantage in predicting or biasing the output values.** Until the hash function is defined, the future hash output of any particular input cannot be predicted any better than guessing the output of a uniform selection over the range.

- **Sampling from random quantum circuits:**

1. **Member of a large class.** The random quantum circuit is (pseudo-)uniformly selected from a large class of circuits, all with the same output range.
2. **Exponential distribution of QC-values.** For each circuit, the set of QC-values follows an exponential model. Contrarily to universal hashing, the adversary \mathcal{A} is given the circuit specification \mathcal{C} , but we assume \mathcal{A} has to publish a sample of strings before being able to compute the corresponding QC-values. Since the circuit class is very large (e.g., exponential in the square of n), we assume a computationally bounded adversary cannot gain any advantage from pre-computations.
3. **Advantage in predicting or biasing the QC-values.** Within the time allowed for publishing a sample, an adversary cannot:
 - predict anything about the QC-value of any string s , any better than a third party could without the circuit specification and who is only told: (i) how many times the string s has been output by quantum evaluation; and (ii) a tally of the repetitions of all evaluations of the circuit (i.e., like m_1 single-time strings; m_2 two-time strings; ...).
 - induce any bias on the SQC, apart from the bias resulting from deciding how many strings to include from each bin (obtained from quantum-sampling) and how many to include without having been output by the circuit.

6. Concluding remarks

We analyzed the sum of probability values (SQC) of the outputs of quantum circuits, and how they can support a certified randomness protocol. The devised parameters hinge on a number of assumptions, such as a computational gap making it infeasible to compute QC-values before the time to publish a sample, but feasible to compute them a posteriori for verification purposes. Under these assumptions, we have calculated, for the case of $n = 53$ qubits, the number m of strings necessary to achieve, in an adversarial setting, various levels of security and efficiency for the generation of certifiable randomness. We have also described how to estimate the amount of entropy that can be certified by these experiments.

The analysis identified some limitations of using the SQC as the base statistic, namely by describing a range of attacks available to an adversary. We have focused on the case of 53 qubits, for which we argued being able to ignore the adversarial advantage of observing collisions. However, this effect would need to be suitably measured for experiments based on fewer qubits. The results obtained here do not directly apply to conceivable approaches based on other statistics (such as Kolmogorov-Smirnov and Chi Square) that take more into account the shape of the distribution of QC values.

We hope these notes motivate the exploration of further relevant aspects. We list here a few items:

1. **Computational complexity.** What are the computational complexities of classically computing a single QC-value vs. the QC-values of sampled strings, vs. the QC-values of all N strings?
2. **Approximate QC-values.** Are there methods to compute approximated QC-values that may yield a verifier speed-up or an adversarial advantage?
3. **Computational gap.** What are safe practical-and-conservative estimates for the concrete complexity gap between quantum evaluation of a random circuit vs. classical evaluation of its QC-values?
4. **Statistical approximations.** We have argued/recommended that in cryptographic settings the false-positive and false-negative rates be set to a very low value, such as 2^{-40} for 40 bits of statistical security. Is the non-exact approximation of the exponential model accurate enough at the tails of the distribution function? In what ranges of parameters can we safely use the CLT and Gamma approximations, and/or what error margins to use, and to which extent is the sampling with replacement well approximated by sampling without replacement?
5. **Entropy estimation.** We have made a few ap-

proximations in the estimation of entropy. It would be interesting to find analytical formulas and/or a formal basis for a more accurate estimation.

6. **Analytic formulas.** We found some formulas by pattern observation (Section D.2), such as for some statistics per bin and per budget factor. They match well our experimental results obtained discretized probability density functions. It would be interesting to obtain a formal derivation of these formulas, and/or corresponding corrections. Some general formulas are yet to be found, for example to include a functional dependency on the fidelity.
7. **Distinguishing statistics.** Can better results of distinguishability be obtained using different statistics, such as Kolmogorov-Smirnov or Chi-Square? (We believe so.) How can these also be applied to the low-budget setting?
8. **Protocols based on PCPs.** Can probabilistically checkable proofs (PCPs) be used to reduce the complexity of the verifier? It would be quite interesting to devise a PCP strategy where the PCP itself could be generated by the quantum computer.
9. **Proofs of security.** We are interested in a cryptographic context and therefore should also aim for corresponding proofs of security. In these notes we have only given intuitive arguments about the classes of adversaries and their capabilities. A more formal analysis is required before actually deciding parameters for real applications.

Apart from having conceived a low-budget method, these notes do not address the issue of whether or not the computational cost of the protocol is reasonable in practice for obtaining the needed assurance of fresh entropy. In any case, this is a step tapping into the potential of quantum randomness applications.

Acknowledgments

We are grateful to Sergio Boixo and Scott Aaronson for explaining their ideas about certifiable randomness using a quantum computer. We initiated this work as a preparation for a presentation at a meeting, held at NIST on December 12–13, 2019, between some members of the “Google AI Quantum” team and the NIST’s “Interoperable Randomness Beacons” team. We thank the feedback by Ray Perlner and Paul Black as NIST WERB reviewers. In an early draft where we considered a predicate-based rejection sampling, Ray suggested we mention the entropy reduction caused by subsequent sorting. Ray also suggested we mention the hash bias attack; we describe how to mitigate it in Section 5.5.

References

- [Aar19] S. Aaronson. *Certified Randomness from Quantum Supremacy*. Unpublished manuscript. 2019.
- [AC16] S. Aaronson and L. Chen. *Complexity-Theoretic Foundations of Quantum Supremacy Experiments*. 2016. arXiv:1612.05903 [quant-ph].
- [AABB+19] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, R. Biswas, S. Boixo, F. G. S. L. Brandao, D. A. Buell, and et al. “Quantum supremacy using a programmable superconducting processor”. In: *Nature* 574.7779 (Oct. 2019), pp. 505–510. doi:10.1038/s41586-019-1666-5. arXiv:1910.11333 [quant-ph].
- [BISB+18] S. Boixo, S. V. Isakov, V. N. Smelyanskiy, R. Babbush, N. Ding, Z. Jiang, M. J. Bremner, J. M. Martinis, and H. Neven. “Characterizing Quantum Supremacy in Near-Term Devices”. In: *Nature Phys* 14 (June 2018), pp. 595–600. doi:10.1038/s41567-018-0124-x. arXiv:1608.00263 [quant-ph].
- [CW77] J. L. Carter and M. N. Wegman. “Universal Classes of Hash Functions (Extended Abstract)”. In: *Proc. 9th Annual ACM Symposium on Theory of Computing*. STOC ’77. 1977, pp. 106–112. doi:10.1145/800105.803400.
- [KBPB19] J. Kelsey, L. T. A. N. Brandão, R. Peralta, and H. Booth. *A Reference for Randomness Beacons: Format and Protocol Version 2. Draft NISTIR 8213*. 2019. doi:10.6028/NIST.IR.8213-draft.
- [NIST] National Institute of Standards and Technology — Computer Security Division. <https://nist.gov/itl/csd>.
- [DLMF] NIST. *NIST Digital Library of Mathematical Functions*. Version 1.0.25 (December 15). Online resource. 2019.
- [PGNHW19] E. Pednault, J. A. Gunnels, G. Nannicini, L. Horesh, and R. Wisnieff. *Leveraging Secondary Storage to Simulate Deep 54-qubit Sycamore Circuits*. 2019. arXiv:1910.09534 [quant-ph].
- [NQIA] U.S.Congress. *National Quantum Initiative Act — Public Law No. 368. 115th Congress (2017-2018) of the United States*. 2018. <https://www.congress.gov/bill/115th-congress/house-bill/6227/text>.

Appendix

A. Terminology

A.1. Abbreviations

- approx.: **approx**imation
- Bin: **Bin**omial (probability distribution)
- e.g.: *exempli gratia* (for example)
- Exp: **Ex**ponential (probability distribution)
- fid: **fid**elity
- Fig.: **Fig**ure

- i.e.: *id est* (that is)
- i.i.d: independently and identically distributed
- MHz: **mega hertz** (one million per second)
- min: **minimum**
- Ref.: (bibliographic) **reference**
- stdev: **standard deviation**
- vs.: *versus* (in contrast to)

A.2. Acronyms

- CDF: **cumulative distribution function**
- CLT: **central limit theorem**
- CSD: **Computer Security Division** (at NIST)
- FN: **false negative rate**
- FP: **false positive rate**
- MHz: **Mega Hertz** (1 million per second)
- NISQ: **noise intermediate-scale quantum** [device]
- NIST: **National Institute of Standards and Technology**
- ORCID: **Open Researcher and Contributor ID**[entifier]
- PCP: **probabilistic checkable proof**
- PDF: **probability density function**
- PRG: **pseudo-random generator**
- PRP: **pseudo-random permutation**
- QC-value: probability of a **quantum-circuit output value**
- SHA: **secure hash algorithm**
- SQC: (statistic) **sum of QC-values**
- STQC: (statistic) **sum of TQC values**
- TQC: (statistic) **truncated QC-value**
- U.S.: **United States**

A.3. Symbols

- #: number of
- !: factorial function (! = $\Gamma(\cdot + 1)$)
- \circ : composition of functions
- \ll : much smaller than
- \approx : approximate
- \equiv : equality by definition
- \sim : equality of probabilistic distributions
- \setminus : set exclusion (e.g., $S_1 \setminus S_2$: elements in S_1 but not in S_2)
- $A_{b,c,\phi}$: expected average QC value of strings in bin c
- b : sampling **budget factor** (β/N)
- β : sampling **budget** (# evaluations by adversary)
- c : multiplicity (or count; # occurrences)
- C : **chosen-count** (sampling type)
- \mathcal{C} : **quantum circuit**
- e : **Euler's number**, Napier's constant (≈ 2.7182818)
- $E+$: (scientific notation) multiplication by power of 10
- $E[\cdot]$: **expected value** of a random variable \cdot
- $\text{erf}(\cdot)$ (**error function**; integral of a Gaussian, from 0 to \cdot)
- ϵ : FP or FN rate

- f : **frequency density**
- F : **fidelity** (sampling type)
- ϕ : **fidelity** (or pseudo-fidelity) value
- $\Phi(\cdot, \cdot, \cdot)$: **Lerch transcendent function**
- γ : Euler-Mascheroni constant (≈ 0.5772157)
- $\gamma(\cdot, \cdot)$: **lower incomplete gamma non-regularized function**
- $\Gamma(\cdot)$: **[Complete] Gamma function**
- h : entropy of a string sampled from a distribution
- $h_{b,c,\phi}$: expected average entropy of strings in bin c
- H : Entropy of a distribution or of a sample of strings
- \mathcal{H} : harmonic number function
- I_k : k -th *moment* of f
- $\log(\cdot)$: natural **logarithm** (base $e \approx 2.71828$) of \cdot
- $\log_2(\cdot)$: **logarithm base 2** of \cdot
- m : **number** of strings in published sample
- $M_{b,c,\phi}$: expected # strings with multiplicity c
- μ : **mean** of a sample
- ν : proportion (out of m) of client-side verified strings
- N : **number** (2^n) of n -bit strings
- \mathcal{N} : Normal curve (Gaussian)
- p : **probability** (often as a mute variable in an integral)
- $P(\cdot)$: lower incomplete gamma regularized function
- q : # strings from correct **quantum circuit evaluation**
- Q : pure **quantum** (sampling type)
- s : a string from within a set (e.g., from S_n)
- S_n : **set** of strings with n bits, e.g., with $n = 53$
- σ : **standard deviation** of a sample
- \sum : **summation** with respect to variable \cdot
- $\int \cdot d\cdot$: **integral** with respect to variable \cdot
- t and τ : **truncation threshold** to count a QC-value
- T : **distinguishability threshold** (to accept vs. reject)
- u : **union** of several bins
- U : **uniform** (sampling type)
- $V[\cdot]$: **variance** of a random variable \cdot
- V_A, V_h, V_M : variances of A, h and M , respectively
- X, Y : random variables

B. Expected value and variance

B.1. Auxiliary primitives

Let $f(x) = N \cdot e^{-N \cdot x}$ and $I_k \equiv \int_{x=0}^1 (f(x) \cdot x^k dx)$, for positive integers N and k . Then:

$$I_0 = 1 - e^{-N} \approx 1 \quad (30)$$

$$I_1 = (1 - e^{-N}(1 + N)) \approx 1/N \quad (31)$$

$$I_2 = (2 - e^{-N}(2 + 2N + N^2)) \approx 2/N^2 \quad (32)$$

$$I_3 = (6 - e^{-N}(6 + 6N + 3N^2 + N^3)) \approx 6/N^3 \quad (33)$$

$$\text{More generally: } I_k \approx (k!)/N^k \quad (34)$$

The approximations only ignore negligible terms (exponentially decreasing) proportional to e^{-N} .

B.2. Expected values

$$E[X_U] = \int_{x=0}^1 (N \cdot f(x) \cdot x \cdot (1/N) dx) = I_1 \approx 1/N \quad (35)$$

$$E[X_Q] = \int_{x=0}^1 (N \cdot f(x) \cdot x \cdot x dx) = N \cdot I_2 \approx 2/N \quad (36)$$

$$\begin{aligned} E[X_{F,\phi}] &= \int_{x=0}^1 (N \cdot f(x) \cdot x \cdot (\phi \cdot x + (1-\phi) \cdot (1/N)) dx) \\ &= \phi \cdot E[X_Q] + (1-\phi) \cdot E[X_U] \approx (1+\phi)/N \end{aligned} \quad (37)$$

When i.i.d. sampling m strings, the expected value of the SQC is simply multiplied by m :

$$E[X_{*,m}] = m \cdot E[X_*]. \quad (38)$$

B.3. Variances

$$\begin{aligned} V[X_U] &= E[X_U^2] - E[X_U]^2 \\ &= \int_{x=0}^1 (N \cdot f(x) \cdot x^2 \cdot (1/N) dx) - E[X_U]^2 \\ &= I_2 - E[X_U]^2 \approx 2/N^2 - 1/N^2 = 1/N^2 \end{aligned} \quad (39)$$

$$\begin{aligned} V[X_Q] &= E[X_Q^2] - E[X_Q]^2 \\ &= \int_{x=0}^1 (N \cdot f(x) \cdot x^2 \cdot x dx) - E[X_Q]^2 \\ &= N \cdot I_3 - E[X_Q]^2 \approx 6/N^2 - (2/N)^2 = 2/N^2 \end{aligned} \quad (40)$$

$$\begin{aligned} V[X_{F,\phi}] &= E[X_{F,\phi}^2] - E[X_{F,\phi}]^2 \\ &= \int_{x=0}^1 (N \cdot f(x) \cdot x^2 \cdot (\phi \cdot x + (1-\phi) \cdot (1/N)) dx) - E[X_{F,\phi}]^2 \\ &= \phi \cdot N \cdot I_3 + (1-\phi) \cdot I_2 - E[X_{F,\phi}]^2 \\ &= \phi \cdot 6/N^2 + (1-\phi) \cdot 2/N^2 - (1+\phi)^2/N^2 \\ &= (1+\phi \cdot (2-\phi))/N^2 \end{aligned} \quad (41)$$

When i.i.d. sampling m strings, the variance of the SQC is simply multiplied by m :

$$E[V_{*,m}] = m \cdot V[X_*]. \quad (42)$$

B.4. The chosen-count sampling case

A malicious adversary with a quantum computer with fidelity 1 is able to choose the exact number q of strings that are sampled as a correct quantum-circuit evaluation, and obtain the remaining $m - q$ strings by uniform sampling. The corresponding random variable can be represented as $X_{C,m,q} = X_{U,m-q} + X_{Q,q}$. Correspondingly, the statistics for this experiment can be derived from those of the uniform and the pure-quantum experiments, as follows:

$$E[X_{C,m,q}] = E[X_{U,m-q}] + E[X_{Q,q}] = (m+q)/N \quad (43)$$

$$V[X_{C,m,q}] = V[X_{U,m-q}] + V[X_{Q,q}] = (m+q)/N^2 \quad (44)$$

For $\phi \notin \{0,1\}$, it is worth noticing that $E[X_{C,m,\phi}] = E[X_{F,m,\phi}]$, but $V[X_{C,m,\phi}] < V[X_{F,m,\phi}]$.

C. Sum of QC-Values (SQC)

In this section we derive formulas for the CDF of the sum of QC values (SQC). A main goal is to determine the number of strings needed to sample to achieve defined goals of FN (false negative) and FP (false positive) for distinguishability experiments based on the SQC statistic.

C.1. CLT approximation

The distribution of SQCs (Sum of QC-values) can be approximated as a normal distribution (\mathcal{N}) with mean μ and standard deviation (stdev) σ , when QC-values are i.i.d.

False Negatives (FN). Let us consider that the honest/reference experiment follows a normal distribution with mean μ_1 and stdev σ_1 . Then, for any given threshold t of the SQC, the false negative rate (FN) ϵ_1 (i.e., that an honest sample is rejected) is given by the integral (45) of the Normal PDF from minus infinity up to the threshold value.

$$\epsilon_1 = \int_{x=-\infty}^t \mathcal{N}[\mu_1, \sigma_1](x) = \frac{1}{2} \cdot \left(1 + \operatorname{erf} \left(\frac{t - \mu_1}{\sqrt{2} \cdot \sigma_1} \right) \right) \quad (45)$$

Solving (45) with respect to the threshold gives:

$$t = \mu_1 - \sqrt{2} \cdot \sigma_1 \cdot \operatorname{erf}^{-1}(1 - 2 \cdot \epsilon_1), \quad (46)$$

where μ_1 and σ_1 can be obtained, respectively as $E(X)$ and $\sqrt{V(X)}$, from the ‘‘Fidelity ϕ ’’ row in Table 3 in Section 3.1.

False Positives (FP). Consider now a malicious sampling characterized by a normal distribution with mean μ_2 and stdev σ_2 . Then, for any given threshold t of the SQCs, the false positive rate (FP) ϵ_2 (i.e., that the malicious sample is accepted) is given by the integral (47) from the threshold value t up to infinity.

$$\epsilon_2 = \int_{x=t}^{\infty} \mathcal{N}[\mu_2, \sigma_2](x) = \frac{1}{2} \left(1 - \operatorname{erf} \left(\frac{t - \mu_2}{\sqrt{2} \cdot \sigma_2} \right) \right) \quad (47)$$

Solving (47) in order of t gives:

$$t = \mu_2 + \sqrt{2} \cdot \sigma_2 \cdot \operatorname{erf}^{-1}(1 - 2 \cdot \epsilon_2) \quad (48)$$

Needed number of strings. To simplify the notation, consider the variables μ'_i and σ'_i defined by: $\mu_i = \mu'_i \cdot (m/N)$ and $\sigma_i = \sigma'_i \cdot (\sqrt{m}/N)$.

Since the threshold t is the same when measuring FN (ϵ_1) and FP (ϵ_2), we can equate (46) and (48).

$$\begin{aligned} m \cdot \mu'_1 - \sqrt{2} \cdot \sqrt{m} \cdot \sigma'_1 \cdot \operatorname{erf}^{-1}(1 - 2 \cdot \epsilon_1) &= \\ m \cdot \mu'_2 + \sqrt{2} \cdot \sqrt{m} \cdot \sigma'_2 \cdot \operatorname{erf}^{-1}(1 - 2 \cdot \epsilon_2) &= \end{aligned} \quad (49)$$

Solving with respect to the number m of strings gives:

$$m = 2 \cdot \left(\frac{\sigma'_1 \cdot \operatorname{erf}^{-1}(1 - 2 \cdot \epsilon_1) + \sigma'_2 \cdot \operatorname{erf}^{-1}(1 - 2 \cdot \epsilon_2)}{\mu'_1 - \mu'_2} \right)^2 \quad (50)$$

The mean μ and the standard deviation σ are functions of the number m of samples, of the fidelity ϕ and of the string space size N . When the acceptance criterion is to have the SQC statistic greater than the defined threshold t , it follows that the distinguishability only makes sense when $\phi_2 < \phi_1$ and $\epsilon \leq 0.5$.

Example 8. Let FN=FP and consider an arbitrary ϕ_2 . Consider the malicious pseudo-fidelity case, where the adversary chooses the number $q = m \cdot \phi_2$ quantum evaluations of the circuit, for some pseudo-fidelity ϕ_2 . Then the expected SQC satisfies $\mu'_2 = (1 + \phi_2)$ (the same as for X_{F,m,ϕ_2}) and the normalized stdev satisfies $\sigma'_2 = \sqrt{1 + \phi_2}$ (which is slightly smaller than for X_{F,m,ϕ_2}). The general honest case has $\mu'_1 = 1 + \phi_1$ and $\sigma'_1 = \sqrt{1 + \phi_1 \cdot (2 - \phi_1)}$. Consider also setting the FN and FP rates to be equal to the same value ϵ . Then replacing the parameters in equation (50) yields:

$$m = 2 \cdot \left(\frac{\operatorname{erf}^{-1}(1-2\epsilon)}{\phi_1 - \phi_2} \right)^2 \cdot \left(\sqrt{1 + \phi_1 \cdot (2 - \phi_1)} + \sqrt{1 + \phi_2} \right)^2 \quad (51)$$

Example 9. Let FN=FP and consider $\phi_2 = 0$. Set again both false rates (FP and FN) to be equal (ϵ). Consider that the honest case has fidelity $\phi_1 = \phi$ and the malicious reference uses $\phi_2 = 0$ (uniform sampling), such that their means μ and standard deviations σ respectively satisfy ($\mu'_1 = 1 + \phi$, $\mu'_2 = 1$) and ($\sigma'_1 = \sqrt{1 + \phi \cdot (2 - \phi)}$, $\sigma'_2 = 1$). Then equation (50) is simplified to:

$$m = \frac{2}{\phi^2} \cdot \left(1 + \sqrt{1 + \phi \cdot (2 - \phi)} \right)^2 \cdot \left(\operatorname{erf}^{-1}(1 - 2 \cdot \epsilon) \right)^2 \quad (52)$$

Sometimes in this setting it can be useful to compute the symmetric rates FN=FP from the number m of strings and from the honest fidelity ϕ . The result is then:

$$\epsilon = \frac{1}{2} \cdot \left(1 - \operatorname{erf} \left(\frac{\phi \cdot \sqrt{m}}{1 + \sqrt{1 + \phi \cdot (2 - \phi)}} \right) \right) \quad (53)$$

C.2. Exact Gamma distributions

In this section we find exact (albeit more complex) formulas for the distribution of sum of QC values. This enables computing exact results in some cases, and to estimate error margins in others, namely as compared to the CLT-Normal approximation (Section C.1). We start with the simplest cases: (i) uniform; (ii) pure quantum; (iii) exact proportions.

C.2.1. SQC under uniform string sampling

Under a uniform string sampling, the model of QC-values postulates an exponential distribution with rate N . The corresponding sum of m i.i.d. variables X_U is a new variable

$X_{m,U}$ with an Erlang distribution with shape m and rate N .

$$\text{PDF}[X_{m,U}](x_m) = N^m \cdot x_m^{m-1} \cdot e^{-N \cdot x_m} / \Gamma(m) \quad (54)$$

$$\text{CDF}[X_{m,U}](x_m) = P(m, N \cdot x_m) \quad (55)$$

Above, Γ is the [complete] Gamma function, satisfying $\Gamma(m) = (m-1)!$ (the “!” represents the factorial function) when m is a positive integer. The function P is the [lower] incomplete gamma regularized function, satisfying:

$$P(a, z) = \gamma(a, z) / \Gamma(a) \quad (56)$$

$$\gamma(a, z) = \int_0^z t^{a-1} \cdot e^{-t} dt \quad (57)$$

where γ is the [lower, non-regularized] incomplete gamma function [DLMF].

C.2.2. SQC under quantum string sampling

Under the evaluation (with fidelity 1) of a random quantum circuit, the model of QC-values of the obtained strings is an Erlang distribution with shape 2 and rate N . The corresponding sum of m i.i.d. variables X_Q is a new variable $X_{m,Q}$ with an Erlang distribution with shape $2m$ and rate N .

$$\text{PDF}[X_{m,Q}](x_m) = N^{2m} \cdot x_m^{2m-1} \cdot e^{-N \cdot x_m} / \Gamma(2m) \quad (58)$$

$$\text{CDF}[X_{m,Q}](x_m) = P(2m, N \cdot x_m) \quad (59)$$

It is useful to notice that one quantum sampling contributes to SQC exactly as two uniform samplings, so $X_{m,Q} \sim X_{2m,U}$.

C.2.3. SQC under pseudo-fidelity sampling

In the pseudo-fidelity case, we assume that an adversary with a quantum computer with fidelity 1 chooses in advance how many (q) strings to sample correctly (quantumly) and how many ($m-q$) to sample uniformly. The pseudo-fidelity is then the proportion $\phi' = m/q$ of quantumly obtained strings.

The sum of $m-q$ i.i.d. variables X_U from uniform sampling, plus the sum of q i.i.d. variables X_Q from quantum sampling, is a new variable $X_{C,m,q}$. By the same argument as in Section C.2.2, where a quantum sampling is like two uniform samplings, we immediately deduce $X_{C,m,q} \sim X_{m+q,U}$. The corresponding CDF, which will be useful ahead to determine false positive rates, is:

$$\text{PDF}[X_{C,m,q}](x_m) = \frac{N^{m+q} \cdot x_m^{m+q-1} \cdot e^{-N \cdot x_m}}{\Gamma(m+q)} \quad (60)$$

$$\text{CDF}[X_{C,m,q}](x_m) = P(m+q, N \cdot x_m) \quad (61)$$

C.2.4. SQC under honest sampling with fidelity

The honest fidelity case is more contrived, since there is a mix of probabilities of a string being obtained uniformly vs. quantumly. Even for a single sample, the random variable

QC-value $X_{F,\phi}$ has a PDF that depends on the fidelity ϕ :

$$\text{PDF}[X_{F,\phi}] = (1 - \phi) \cdot \text{PDF}[X_U] + \phi \cdot \text{PDF}[X_Q] \quad (62)$$

$$= (1 - \phi) \cdot \text{Exp}[N](x) + \phi \cdot \text{Erlang}[2, N](x) \quad (63)$$

$$= (1 - \phi) \cdot N \cdot e^{-N \cdot x} + \phi \cdot N^2 \cdot x \cdot e^{-N \cdot x} \quad (64)$$

$$= ((1 - \phi) + \phi \cdot N \cdot x) \cdot N \cdot e^{-N \cdot x} \quad (65)$$

Let $Z_{m,\phi}$ be the random variable denoting the number of strings that are selected, out of m trials, for quantum evaluation in a fidelity ϕ experiment. $Z_{m,\phi}$ has a Binomial distribution $\text{Bin}[m, \phi]$, with parameters m (“number of trials”) and p (“probability of success per trial”). This is a discrete distribution, with PDF:

$$\text{PDF}[Z_{m,\phi}](q) = \binom{m}{q} \cdot \phi^q \cdot (1 - \phi)^{m-q}, \quad (66)$$

where $\binom{m}{q}$ is the binomial coefficient m choose q , satisfying:

$$\binom{m}{q} = \frac{m!}{q! \cdot (m - q)!}, \quad (67)$$

for any integer $q \in \{0, \dots, m\}$.

PDF of SQC. The SQC random variable $X_{m,F,\phi}$ can now be represented as a mix of cases of pseudo-fidelity, when taking in consideration the probabilities of each possible selected number of quantum evaluations. More concretely:

$$\begin{aligned} \text{PDF}[X_{m,F,\phi}](x_m) &= \\ &= \sum_{q=0}^m \text{PDF}[\text{Bin}[m, \phi]](q) \cdot \text{PDF}[X_{E,m,q}](x_m) \end{aligned} \quad (68)$$

$$= \sum_{q=0}^m \binom{m}{q} \cdot \phi^q \cdot (1 - \phi)^{m-q} \cdot \frac{N^{m+q} \cdot x_m^{m+q-1}}{\Gamma(m+q) \cdot e^{N \cdot x_m}} \quad (69)$$

$$= \left[\sum_{q=0}^m \binom{m}{q} \cdot \frac{(\phi \cdot N \cdot x_m)^q}{\Gamma(m+q)} \right] \cdot \frac{(1 - \phi)^m \cdot N^m \cdot x_m^{m-1}}{e^{N \cdot x_m}} \quad (70)$$

$$= \left[\sum_{q=0}^m \binom{m}{q} \frac{(\alpha \cdot x_m)^q}{\Gamma(m+q)} \right] \cdot \frac{\text{PDF}[X_{m,U}](x_m) \cdot \Gamma(m)}{(1 - \phi)^{-m}} \quad (71)$$

The sum can be expressed as a [\(non-regularized\) Kummer confluent hypergeometric function](#), but what we are really interested in evaluating is the CDF and its inverse.

CDF of SQC. The CDF can be obtained by the integral of the PDF, from 0 till the value x_m of interest. Alternatively, and analogously to how the PDF was defined as a weighed (Binomial) sum of PDFs (68), we can also obtain the CDF as a sum (73) of regularized incomplete gamma functions. This follows from the sum with respect q being commutative with the integral (transforming the PDF into the CDF) with respect to x .

$$\begin{aligned} \text{CDF}[X_{m,F,\phi}](x_m) &= \\ &= \sum_{q=0}^m \text{PDF}[\text{Bin}[m, \phi]](q) \cdot \text{CDF}[X_{E,m,q}](x_m) \end{aligned} \quad (72)$$

$$= (1 - \phi)^m \cdot \sum_{q=0}^m \binom{m}{q} \cdot \left(\frac{\phi}{1 - \phi} \right)^q \cdot P(m + q, N \cdot x_m) \quad (73)$$

We can do some operations to move the gamma function outside of the sum. Let $t \equiv N \cdot x_m$. First we obtain $P(m + q, t)$ as a function of $P(m, t)$, by using a recurrence relation:

$$P(m + q, t) = P(m + q - 1, t) - e^{-t} \cdot \frac{t^{m+q-1}}{m + q - 1} \quad (74)$$

$$= P(m, t) - e^{-t} \cdot \sum_{j=1}^q \frac{t^{m+q-j}}{m + q - j} \quad (75)$$

$$= P(m, t) - e^{-t} \cdot \sum_{j=0}^{q-1} \frac{t^{m+j}}{m + j} \quad (76)$$

The last sum can be expressed in terms of the Lerch transcendent function Φ (do not confuse with fidelity ϕ), as follows:

$$\sum_{j=0}^{q-1} \frac{t^{m+j}}{m + j} = t^m \cdot \Phi(t, 1, m) - t^q \cdot \Phi(t, 1, m + q) \quad (77)$$

Inserting the result into (73), and letting $\alpha = \phi/(1 - \phi)$ gives:

$$\begin{aligned} \text{CDF}[X_{m,F,\phi}](x_m) &= \\ &= P(m, t) - e^{-t} \cdot \left(\frac{t^m \cdot \Phi(t, 1, m)}{(1 - \phi)^m} - \sum_{q=0}^m \binom{m}{q} \cdot \frac{\Phi(t, 1, m+q)}{\alpha^{-q}} \right). \end{aligned} \quad (78)$$

While (78) does not directly hint at an easier way to compute the CDF of the general fidelity case, compared with (73), it does however show the contribution that the quantum fidelity ϕ brings to the CDF of the SQC. Recall that the SQC of the uniform experiment (fidelity 0) is equal to $P(m, t)$.

C.2.5. Needed sample size (number of strings)

False Negatives (FN). When measuring SQC values of m i.i.d. sampled strings from an honest distribution with fidelity ϕ , the false negative rate (FN) ϵ_1 is given by the CDF of the distribution evaluated at the threshold t :

$$\epsilon_1 = \text{CDF}[X_{m,F,\phi}](t) \quad (79)$$

Solving with respect to t gives:

$$t = \text{CDF}^{-1}[X_{m,F,\phi}](\epsilon_1) \quad (80)$$

For practical purposes, we can sometimes use the approximation obtained in (93).

False Positives (FP). For a threshold t of the SQC values of m i.i.d. maliciously sampled strings with pseudo-fidelity ϕ_2 , i.e., with exactly $q = m \cdot \phi_2$ quantum evaluations, the false positive rate (FP) ϵ_2 is (based on (61)):

$$\epsilon_2 = \text{CDF}[X_{E,m,\phi_2,m}](x_m) = P(m + q, N \cdot t) \quad (81)$$

Solving with respect to t gives:

$$t = N^{-1} \cdot P_{(\cdot, m+q)}^{-1,1}(\epsilon_2), \quad (82)$$

where $P_{(\cdot, m+q)}^{-1,1}$ denotes the inverse of P with respect to the first argument, when the second argument is $m+q$. In the simple uniform case ($\phi_2 = 0$) the equation becomes:

$$t = N^{-1} \cdot P_{(\cdot, m)}^{-1,1}(\epsilon_2) \quad (83)$$

Solving for the sample size. Given target FP and FN rates (ϵ_1 and ϵ_2) for a distinguishability experiment, the needed number m of strings is given by equating (80) and (82), thus obtaining:

$$\text{CDF}^{-1}[X_{m,F,\phi}](\epsilon_1) = N^{-1} \cdot P_{(\cdot, m \cdot (1+\phi_2))}^{-1,1}(\epsilon_2) \quad (84)$$

and then solving in order of m .

C.3. A Gamma approximation

Since the exact CDF (73) for the honest fidelity case is complicated (i.e., at least difficult to evaluate as an explicit Binomial sum, for very large m), we consider an approximation obtained by means of a pseudo-fidelity sampling ($X_{C,m,m,\phi'}$) with parameters (N' , ϕ') adjusted to match the expected value and the variance of the honest fidelity case ($X_{m,F,\phi}$).

Recall here the two expected values and two variances:

$$E[X_{C,m,m,\phi'}] = m \cdot (1 + \phi')/N' \quad (85)$$

$$V[X_{C,m,m,\phi'}] = m \cdot (1 + \phi')/N'^2 \quad (86)$$

$$E[X_{m,F,\phi}] = m \cdot (1 + \phi)/N \quad (87)$$

$$V[X_{m,F,\phi}] = m \cdot (1 + \phi \cdot (2 - \phi))/N^2 \quad (88)$$

We can equate the statistics of interest:

$$\frac{m \cdot (1 + \phi')}{N'} = \frac{m \cdot (1 + \phi)}{N} \quad (89)$$

$$\frac{m \cdot (1 + \phi')}{N'^2} = \frac{m \cdot (1 + \phi \cdot (2 - \phi))}{N^2} \quad (90)$$

Then, solving with respect to N' and ϕ' gives:

$$N' = N \cdot \frac{1 + \phi}{1 + \phi \cdot (2 - \phi)} \quad (91)$$

$$\phi' = \phi \cdot \frac{2 \cdot \phi}{1 + \phi \cdot (2 - \phi)} \quad (92)$$

The SQC variable $X_{[N']m,F,\phi'}$ of honest fidelity sampling is approximately the same as the SQC variable $X_{[N']C,m,m,\phi'}$ of pseudo-fidelity sampling with parameters N' and ϕ' adjusted as in (91) and (92), respectively. Correspondingly, the CDF and PDF are as follows:

$$\text{CDF}[X_{[N']m,F,\phi'}] \approx P(m \cdot (1 + \phi'), N' \cdot x) \quad (93)$$

$$\text{PDF}[X_{[N']m,F,\phi'}] \approx N'^{1+\phi'} \cdot x^{\phi'} \cdot e^{-N' \cdot x} / \Gamma(1 + \phi') \quad (94)$$

This provides a better approximation than the Normal-CLT, while also allowing an easy formulation of the sum of i.i.d. variables. Any discrepancy found between the CLT-Normal and this case is on its own evidence of a lack of applicability of the CLT, since the Gamma approximation has the same mean and variance as the non-approximated case.

Table 10: Gamma vs. Normal (CLT) approximations

| ϕ | m | $\log_2(\text{FN}_{\phi_1=\phi})$ | | $\log_2(\text{FP}_{\phi_2=0})$ | |
|--------|----------------|-----------------------------------|-------|--------------------------------|-------|
| | | Gamma | CLT | Gamma | CLT |
| 0.002 | 10^6 | -2.7 | -2.7 | -2.7 | -2.7 |
| | 10^7 | -10.3 | -10.3 | -10.3 | -10.3 |
| | $4 \cdot 10^7$ | -32.8 | -32.8 | -32.9 | -32.9 |
| 0.01 | 10^5 | -4.1 | -4.1 | -4.1 | -4.1 |
| | 10^6 | -21.4 | -21.4 | -21.7 | -21.7 |
| | $2 \cdot 10^6$ | -39.6 | -39.5 | -40.1 | -40.2 |
| 0.05 | 10^4 | -6.9 | -6.9 | -7.3 | -7.3 |
| | $5 \cdot 10^4$ | -24.7 | -24.3 | -26.0 | -26.4 |
| | 10^5 | -46.0 | -45.3 | -48.7 | -49.4 |
| 0.25 | 10^3 | -11.5 | -11.0 | -13.8 | -14.7 |
| | $2 \cdot 10^3$ | -20.4 | -19.3 | -24.7 | -26.4 |
| | $4 \cdot 10^3$ | -37.6 | -35.4 | -46.0 | -49.4 |
| 1 | 10 | -3.0 | -2.9 | -3.8 | -4.1 |
| | 10^2 | -14.1 | -12.3 | -17.4 | -21.7 |
| | $3 \cdot 10^2$ | -36.6 | -31.0 | -45.4 | -58.6 |

Threshold: For illustration purposes, the SQC threshold for measuring FN and FP is set to $(1 + 0.5 \cdot \phi) \cdot m/N$, which is exactly in between the expected values $(1 + \phi) \cdot m/N$ of the SQC in the honest case and m/N in the malicious uniform case. This is a crude (not optimal) approximation of a threshold providing $\text{FN} \approx \text{FP}$.

Legend: ϕ (fidelity in the honest case); m (number of sampled strings used in the SQC); FN and FP (false positive and false negative rates, respectively).

C.3.1. Accuracy analysis

It is instructive to compare results between the Gamma and the CLT approximations. We expect the discrepancies to be more noticeable for not-too-large sample sizes (e.g., $m < 10^3$) or in the tail of the distributions, i.e., when using thresholds for very low FP and FN rates, e.g., of the order of 2^{-40} .

In Section 3.2 we already compared curves for $m = 1$ (Fig. 5) and $m = 5$ (Fig. 6). Table 10 shows other concrete results of FN and FP rates between the exact/approximated gamma calculation and the Normal-CLT approximation. In the Gamma column for the FN case, when $m < 5 \cdot 10^3$ we use the exact weighted binomial sum of Gamma CDFs (73), and otherwise use the derived Gamma-approximation (93). The results in the Gamma column for FP rates are calculated without approximation (i.e., apart from rounding precision), since for pseudo-fidelity 0 the formula is already exact.

Since we do not yet report a numeric solving of (84) with respect to m , we used an heuristic threshold — the middle point SQC between the mean of the honest and the mean of the malicious one. Some observations:

- For $\phi = 1$ and $m = 10^2$, the CLT-Normal FP $\approx 2^{-21.73}$ is about 20 times lower than the Gamma FP $2^{-17.36}$. For $m = 3 \cdot 10^2$ the discrepancy is even larger, e.g., of about 48 times for the FN results. The Gamma results are exact here, since they refer to fidelity either 0 (for FP) or 1 (for FN). This illustrates the need to obtain estimates better than what the CLT-approximation allows.
- For $\phi = 0.25$ and $m = 4 \cdot 10^3$, the CLT-Normal FP

$\approx 2^{-49.42}$ is about 11 times lower than the Gamma FP $2^{-45.98}$. The Gamma result is exact here, since it is calculating the case of fidelity 0.

- For $\phi \in \{0.002, 0.01\}$, the shown CLT rates are very close to the Gamma ones, within one decimal of the \log_2 . This is expected, since these cases involve many sampled strings ($m \geq 10^5$).

D. Discretization of QC-values

D.1. Individual probabilities

Based on the exponential model, we consider how to discretize the QC values. First, we compute the n sequential intervals expected, each, to contain exactly one QC-value. Recall that the CDF $F(p)$ of the frequency-density f gives the number of strings with QC-value up to the input. For example, $F(1/N) \approx 0.632$ means that a factor of about 63.2 % of the strings have QC-value up to $1/N$. Thus, we compute the values x_i such that $F(x_i) = i/N$, for all $i \in \{0, 1, \dots, N\}$. The intuition is that between x_{i-1} and x_i we expect to encounter the i^{th} QC-value p_i . We get:

$$x_i = \frac{\log(N) - \log(N - i)}{N}. \quad (95)$$

The extremes of (95) are $x_0 = 0$ and $x_N = \infty$. The before-last interval ends in $x_{N-1} = \log(N)/N$, which is thus a lower approximation of the maximum QC-value.

If we position all N QC-values to be the minimum value in the interval, i.e., $p_i = x_{i-1}$, then their sum becomes close to, but not exactly, 1.

$$\sum_{i=0}^{N-1} x_i = \log(N) - \frac{\log(\Gamma(N+1))}{N} \quad (96)$$

To enable the sum of QC-values to be 1, we let each p_i be at a more advanced position in the interval $[x_{i-1}, x_i]$. Concretely, we let $p_i = x_{i-1} + \Delta_i$, such that the integral under the frequency-density curve leads to a certain constant factor c , with $0 < c < 1$, of the total $1/N$, i.e.,

$$\int_{x_i}^{x_i + \Delta_i} f(x) dx = \frac{c}{N} \quad (97)$$

Solving in order of Δ_i yields

$$\Delta_i = -\frac{1}{N} \cdot (\log(N - i) - \log(N - i - c)). \quad (98)$$

The sum of QC-values is now given as:

$$\left(\sum_{i=0}^{N-1} x_i + \Delta_i \right) = \log(N) - \frac{1}{N} \cdot \log\left(\frac{\Gamma(N+1-c)}{\Gamma(1-c)}\right). \quad (99)$$

For each N , we set c to the value that makes the sum of probabilities be 1. This value c gets closer to $1/2$ as N increases. For example, for $N = 2^{53}$ we have $c \approx 0.50895$.

A different criterion for discretization could be to let each

initial p_i approximation enforce the Shannon entropy to be equal to the differential entropy, i.e., satisfying:

$$\int_{x_{i-1}}^{x_i} N \cdot f(x) \cdot x \cdot \log_2(x) dx = p_i \cdot \log_2(p_i) \quad (100)$$

And then adapt the p_i values so that they all sum to 1.

D.2. Collisions

An adversary with a large quantum-sampling budget β can use the frequency of collisions of each quantumly-obtained string as a hint about the high or low QC-value of the string. Such capability can be used to affect the sum of QC-values.

When quantumly sampling $\beta = b \cdot N$ strings (for some *budget factor* $b > 0$), with replacement, the adversary may obtain repeated strings. The adversary partitions the set into bins of different multiplicities. Each multiplicity c denotes the number of times a string appears ($c = 1$ means appearing without repetition; higher c values mean actual collisions). We also consider the case of multiplicity $c = 0$, denoting the bin of strings that did not appear. For each bin, we are interested in the expected values of the following measures:

- M : Number of strings
- A : Average of QC-values
- h : Average entropy per string (as measured with respect to an appropriate distribution — see Section D.2.2).

Note that the number of strings can also be seen as a normalized (multiplied by N) average (across all strings) of the probability of occurrence in a bin. Besides averages, sometimes it can also be useful to know the corresponding variances.

For the purpose of the rejection sampling in Section 5, we are interested in uniform sampling across a bin or union of bins.

D.2.1. Initial statistics per bin

We start by considering the probabilities of a string appearing with multiplicity c , when sampling with budget factor b . If a string has QC-value p , then the probability that it is output in an individual circuit evaluation with fidelity ϕ is $p' = \phi \cdot p + (1 - \phi)/N$. When evaluating β times, the probability p'' that the string appears exactly c times is:

$$p'' = \binom{\beta}{c} \cdot p'^c \cdot (1 - p')^{\beta - c}, \quad (101)$$

(see (67) for an explanation of the Binomial notation).

Using p'' as an abbreviation for $p_{N,b,c,\phi}$, letting $k = \sum p''$ be the normalization factor for a corresponding (discrete) probability density function, and using p as an abbreviation of QC-value, we can present formulas for determining the mentioned statistics of interest:

$$M_{N,b,c,\phi} = \sum p'' \equiv k \quad (102)$$

$$A_{N,b,c,\phi} = \sum \frac{p''}{k} \cdot p = \frac{1}{k} \cdot \sum p'' \cdot p \quad (103)$$

$$\begin{aligned}
h_{N,b,c,\phi} &= \sum \frac{p''}{k} \cdot \log_2 \left(\frac{p''}{k} \right) \\
&= - \left(\frac{1}{k} \cdot \sum p'' \cdot \log_2(p'') \right) + \log_2(k) \quad (104) \\
&\quad (105)
\end{aligned}$$

where the summations are over all strings. The formulas are meaningful for non-negative multiplicities. The bin $c = 0$ contains the strings that do not appear while sampling.

Note: p'' is the probability that a string (the index remains implicit) appears in bin c , rather than its probability density in the PDF vector corresponding to bin c . The latter probability density can be obtained by normalization, becoming p''/k . That is for example the reason for the term $\log_2(k)$ in the formula for the entropy.

The list of formulas can be naturally extended to include higher moments of M , A and h . For example, for the variances we have:

$$V_{M,N,b,c,\phi} = \sum p''^2 - k^2 \quad (106)$$

$$V_{A,N,b,c,\phi} = \frac{1}{k} \cdot \sum p'' \cdot p^2 - A_{N,b,c,\phi}^2 \quad (107)$$

$$V_{h,N,b,c,\phi} = \frac{1}{k} \cdot \sum p'' \cdot \log_2 \left(\frac{p''}{k} \right)^2 - h_{N,b,c,\phi}^2 \quad (108)$$

We can now analyze result for several concrete cases. We have experimentally observed that the statistics M , A and V_A closely match what the following formulas predict. We hypothesize that the formulas are correct approximations, but present them here only as a heuristic result, since we obtained them from observing patterns, and have not yet done a proper derivation.

Concrete results for M , A , V_A

- For uniform sampling (fidelity $\phi = 0$):

$$M_{N,b,c,\phi=0} \approx N \cdot \frac{b^c}{e^b \cdot c!} \quad (109)$$

$$A_{N,b,c,\phi=0} = \frac{1}{N} \quad (110)$$

$$V_{A,N,b,c,\phi=0} = \frac{1}{N^2} \quad (111)$$

- For quantum evaluation with fidelity $\phi = 1$:

$$M_{N,b,c,\phi=1} = N \cdot \frac{b^c}{(1+b)^{1+c}} \quad (112)$$

$$A_{N,b,c,\phi=1} = \frac{1}{N} \cdot \frac{1+c}{1+b} \quad (113)$$

$$V_{A,N,b,c,\phi=1} = \frac{1}{N^2} \cdot \frac{1+c}{(1+b)^2} \quad (114)$$

Two identities. Based on the semantics of the expected number $M_{b,c,\phi}$ of strings for each multiplicity c , when sampling with fidelity ϕ and a budget factor b , we can express two useful and straightforward identities:

- **String space size** — it is equal to the sum of frequencies

across all non-negative multiplicities:

$$\sum_{c=0}^{\infty} M_{N,b,c,\phi} = N \quad (115)$$

- **Sampling budget** — it is equal to the sum of the frequencies times multiplicities:

$$\sum_{c=1}^{\infty} c \cdot M_{N,b,c,\phi} = b \cdot N \quad (116)$$

It is straightforward to check that the formulas (112, 109) presented for the frequencies $M_{b,c,\phi}$, when $\phi = 1$ and $\phi = 0$, satisfy the mentioned identities.

D.2.2. Concrete results for entropy

By [apriori] *entropy of a string in bin c* we mean the entropy of the string conditioned on it being in bin c , before rejection sampling. For fidelity $\phi = 0$, strings are obtained uniformly, implying that, within each bin, each string is equally likely to appear. Therefore, regardless of the budget factor b , the uniform distribution remains the basis to compute the expected entropy per string in any bin c , yielding

$$h_{N,b,c,\phi=0} = \log_2(N). \quad (117)$$

The case is more complex when considering a positive fidelity. Table 16 shows, for several budget factors b , the decrease (compared to n) in the apriori entropy in each bin c , as calculated numerically for the case of 28 qubits. The entropy-decrease ($n - h$) increases with the number n of qubits, but the shown values are reasonably close across a wide range of values for n . For example, if using these values for the case of 53 qubits, and assuming a sampling budget of $2^{32} = 2^{-21} \cdot N$, we can confidently say that an adversary that uses a string from bin $c = 2$ is effectively inducing an expected entropy of no more than $51.34 \approx 53 - 1.6626$ bits (before rejection sampling). This is already a noticeable correction from the 52.39 bits determined for the sampling of a single string from the QC-value distribution.

Note: The calculated entropy is, as expected, smaller than what would be (incorrectly) obtained if measuring it directly with respect to the distribution of QC-values. For example, in the case $\phi = 1$ then we would (incorrectly) compute the average \log_2 of the QC-values in each bin c , yielding

$$h_{N,b,c,\phi=1}^* \approx \log_2(N) + \log_2(1+b) + \frac{\gamma - \mathcal{H}(c)}{\log(2)}, \quad (118)$$

where γ is the Euler-Mascheroni constant (≈ 0.577) and $\mathcal{H}(c)$ is the c -th harmonic number ($\sum_{i=1}^c \frac{1}{i}$).

As a clear example of unoptimality, for $(c, b) = (1, 1)$ the formula yields a value larger than n , where n is the value that could already be obtained with a uniform sampling.

D.2.3. Statistics in unions of bins

As an example, consider the union of all bins with positive multiplicity, i.e., corresponding to the set of all obtained distinct strings. Consider a sampling budget of $\beta = b \cdot N$ circuit evaluations. For simplicity of notation, we leave implicit the indices N and b . Consider the use of a prime ($'$) as an identifier that a variable refers to the strings in the union of bins with multiplicity equal to or larger than c (which for $c = 1$ means the set of all distinct sampled strings). Then, the expected number of distinct strings is about:

$$M'_{c=1,\phi} = \sum_{c=1}^{\infty} M_{c,\phi} \quad (119)$$

$$M'_{c=1,\phi=0} = N \cdot (1 - e^{-b}) = N \cdot (1 - e^{-\beta/N}) \quad (120)$$

$$M'_{c=1,\phi=1} = N \cdot \frac{b}{1+b} = N \cdot \frac{\beta}{N+\beta}. \quad (121)$$

Often we also leave implicit the parameter ϕ when clear from the context (e.g., using M'_1 to denote $M'_{c=1,\phi=1}$).

The expected QC value when uniformly selecting a string from within the set of distinct strings is equal to the expected mean QC-value from the set (namely ignoring the information in the tally of collisions) and is equal to:

$$A'_{c=1,\phi} = \frac{1}{M'_{c=1,\phi}} \cdot \sum_{c=1}^{\infty} A_{c,\phi} \cdot M_{c,\phi} \quad (122)$$

$$A'_{c=1,\phi=0} = \frac{1}{N} \quad (123)$$

$$A'_{c=1,\phi=1} = \frac{1}{N} \cdot \frac{2+b}{1+b} \quad (124)$$

This shows that $A'_{c=1,\phi=1}$ is very close to $2/N$ for a budget factor b less than the square-root of the string space N , and it approaches $1/N = A'_{c=1,\phi=0}$ as b tends to infinity (i.e., when all strings are in the pre-sample).

D.3. Approximate sum of “entropies”

Section 5.3.2 provided an initial approximation formula (19) for the expected entropy H of a sample of q strings output by a pseudo-fidelity adversary with budget β , as follows:

$$\underbrace{q \cdot h_{\beta}}_{\text{Apriori entropy}} \underbrace{\left(q \cdot (\log_2(M'_1/q)) + \log_2(q!) \right)}_{\text{entropy reduction}} \quad (125)$$

In this section, we consider corrections to both entropy components: the apriori component; and the reduction component. The second term is an approximation to the entropy removed by rejection sampling; the first term is an approximation to the entropy that would be assigned per string in an experiment that would output the strings by uniform sampling from within the pre-sample.

D.3.1. Apriori entropy

Let *apriori entropy* denote the initial average entropy h_{β} per string in a pre-sample of size M'_1 . Then, a first approximation

for q strings uniformly selected from the sample is obtained by summing q times the average, yielding $q \cdot h_{\beta}$. However, this does not account for the dependencies (that induce a smaller entropy) related to sampling without replacement from the set of M'_1 strings.

Asymptotic behavior. To illustrate the problem, consider the limit when $\beta \rightarrow \infty$ (implying $M'_1 \rightarrow N$). Then all strings are obtained in the pre-sample, meaning that a subsequent uniform selection therefrom has $h_{\infty} = \log_2(N)$. In the specific case $q = N$, i.e., if sequentially sampling all strings, then the overall entropy is 0, since the ordering of all strings is a predictable output. However, $N \cdot \log_2(N) \neq \log_2(N!)$. Thus, the first additive term $q \cdot h_{\beta}$ needs to be an expression that in the limit $\beta \rightarrow \infty$ and $q = M'_1$ converges to $\log_2(N!)$

Sequential correction. Let us say that $h_{\beta_1} \equiv h_{\beta}$ is the entropy of the first string whose entropy is accounted for. Then the second string will require a correction based on the conditional probability distribution that the first string can no longer be selected. As an approximation, let us assume that the entropy of each string in the pre-sample is equal to the average entropy in the set of sampled strings. Then the corrected entropy for the second string becomes:

$$h_{\beta,2} = h_{\beta} + \log_2(1 - 2^{-h_{\beta}}), \quad (126)$$

where $1 - 2^{-h_{\beta}}$ is the renormalization factor to be applied to all probabilities, since $2^{-h_{\beta}}$ was the probability of the string that can no longer appear. Applying this recursively yields:

$$h_{\beta,i} = h_{\beta,i-1} + \log_2(1 - 2^{-h_{\beta,i-1}}) \quad (127)$$

A simpler form of the above recursion is:

$$2^{h_{\beta,i}} = 2^{h_{\beta,i-1}} - 1 = 2^{h_{\beta}} - (i - 1) \quad (128)$$

For the correction of entropy estimation we should now replace $q \cdot h_{\beta}$ by $\sum_{i=1,\dots,q} h_{\beta,i}$, which yields:

$$\sum_{i=1,\dots,q} \log_2(2^{h_{\beta}} - (i - 1)) \quad (129)$$

$$= \log_2(\Gamma(1 + 2^{h_{\beta}})) - \log_2(\Gamma(1 + 2^{h_{\beta}} - q)) \quad (130)$$

$$= \log_2((2^{h_{\beta}})^{\underline{q}}). \quad (131)$$

where $x^{\underline{q}}$ is the descending factorial $x \cdot (x - 1) \cdot \dots \cdot (x - q + 1)$, whose binary logarithm is calculatable as $\log_2(\Gamma(x + 1)) - \log_2(\Gamma(x - q + 1)) / \log(2)$.

Example. $(n, \beta, q) = (53, 2^{32}, 2^{20})$ yields $h_{\beta} \approx 52.4$ and $q \cdot h_{\beta} \approx 5.24\text{E}+7$. The correction yields $(5.24\text{E}+7) + \Delta$, with $\Delta \approx 1.3\text{E}-4$. The variation factor $\Delta / (q \cdot h_{\beta}) \approx 2.4\text{E}-12$ is very small. If using $q = 2^{30}$ that factor is about $2.5\text{E}-9$.

The effect is small and less notorious as the sampling budget factor decreases, which is expected as the number of qubits increases. As an example, consider the case of a collisional adversary in a setting with parameters $(n, \beta, c, h) = (28, 2^n, 16, 11)$ (values from Table 16). Then, the bin $c = 16$ is expected to have about $M_c = 2^{8.5}$ strings. If (for the sake of argument) the adversary would select all of these, then $q \cdot h$ yields about 3982 bits of entropy, whereas the corrected version (before sorting) gives 3933 bits of entropy, i.e., a reduction factor of about $\Delta = 1.23\%$. If instead considering

$(n, \beta, q, h) = (53, 2^{32}, 2^{20}, 52.39)$, then the reduction factor is of only about 2.4E-12.

D.3.2. Entropy upon rejection sampling

Consider the setup of obtaining a pre-sample with M strings from a string space of size N . We focus on rejection sampling techniques that deterministically select a sequence of q strings from the pre-sample.

We look at techniques based on application of a pseudo-random permutation (PRP), which is a bijection parametrized by a secret key. Consider the following examples:

1. **Predicate-based.** Select the strings that upon application of a PRP start with $\lfloor \log_2(M) - \log_2(q) \rfloor$ zeros. We make the simplifying assumption that this selects exactly q strings.
2. **Single-ordering-based.** Apply to every string the same PRP and then select only the lexicographically first q strings, in the corresponding order.
3. **Multi-ordering-based.** Apply a PRP as in the single-ordering-based case, but select only the lexicographically first string, hereafter denoted as s_1 ; then specify another (differently seeded) PRP, mapping the set $S_n \setminus \{s_1\}$ onto itself, and again select the corresponding lexicographically first string s_2 ; ... and so on until selecting q strings.

Different entropy reductions. The distribution of probabilities is distinct across the three cases. The first procedure reduces the number of possible strings (i.e., those with positive probability of selection) by a factor of about $\approx M/q$. The second case only prevents the $M - q$ PRP-lexicographically higher strings from ever being output, although many other strings have a very low (but not 0) probability of occurrence. The third procedure gives a chance to every string (depending on the sequence of PRPs), because strings do not have a fixed lexicographical ranking when considering a sequence of selections.

A toy example. For an illustration of entropy reduction, we first consider a toy example where we can compute the exact entropy reduction. Let $(N, M) = (2^{10}, 2^j)$ for some positive j , such that we intend to obtain a pre-sample with exactly M strings by repeated uniform selection (without replacement) from a set with N strings. Then we do rejection sampling on the M strings in order to select the lexicographically first string. The exact probability of the i -th string being the one selected is

$$p_{N,M}(i) = \frac{\binom{N-i}{M-1}}{\binom{N}{M}} = M \cdot \frac{(N-i)! \cdot (N-M)!}{(N-i-M)! \cdot N!} \quad (132)$$

$$= M \cdot \frac{(N-M)^{\underline{i}}}{N^{\underline{i}}}, \quad (133)$$

where \underline{i} in the exponent denotes the degree of a descending factorial. In the right side of the first equality: the numerator is the number of pre-samples whose lowest value is i ; the denominator is the total number of possible pre-samples.

From the corresponding PDF we can compute the Shannon entropy of the distribution, as the sum of $p \cdot \log_2(p)$ of every

occurring probability p :

$$H_{N,M} = \sum_{i=1}^N p_{N,M}(i) \cdot \log_2(p_{N,M}(i)) \quad (134)$$

For example, we get $H_{2^{10}, 2^1} \approx 9.72$ and $H_{2^{10}, 2^9} \approx 1.998$. These values are noticeably larger than what we would get by applying $H = h - \log_2(M)$ (the special case of $q = 1$ from equation (19)), which would respectively give 9 and 1.

Our approximation — first step ($q = 1$). We start by estimating what is the sampling budget β that induces a pre-sample of size M . For the uniform case this means resolving formula (120) for b , which yields $b = -\log(1 - M/N)$, where $\beta = b \cdot N$. For the case of quantum sampling we would instead solve (121), yielding $b = M/(N - M)$. Now we assume, as an approximation, that whenever β strings are independently sampled (uniformly or quantumly, depending on the used formula), exactly M distinct strings are obtained. Assuming β i.i.d. evaluations to obtain a pre-sample, we know how to calculate the probability $p_i^{(1)}$ of the i -th string being the lexicographically first:

$$p_i^{(1)} = (p_i + z_i)^\beta - (z_i)^\beta, \quad (135)$$

where $z_i = \sum_{k=i+1}^N p_k$ satisfies $z_i = z_{i-1} - p_i$ with $z_0 = 1$. Table 11 compares, for the toy example with uniform selection and $N = 1024$, the exact entropy vs. the one obtained with our proposed approximation vs. the rough approximation obtained from (19).

Table 11: Entropy approximations: Uniform, $q = 1$

Toy example with uniform selection and $(N, q) = (1024, 1)$

| M | H (entropy upon rejection sampling) | | |
|---------------------------|---------------------------------------|-----------------|--------------|
| | Exact | Numeric approx. | Formula (19) |
| $1 = 2^0$ | 10 | 10 | 10 |
| $2 = 2^1$ | 9.72064 | 9.72064 | 9 |
| $4 = 2^2$ | 9.07990 | 9.07990 | 8 |
| $32 = 2^5$ | 6.37554 | 6.37554 | 5 |
| $256 = 2^8$ | 3.24025 | 3.24022 | 2 |
| $512 = 2^9$ | 1.99805 | 1.99797 | 1 |
| $768 = \sum_{i=8}^9 2^i$ | 1.08084 | 1.08070 | 0.41504 |
| $992 = \sum_{i=5}^9 2^i$ | 0.20693 | 0.20679 | 0.04580 |
| $1020 = \sum_{i=2}^9 2^i$ | 0.03699 | 0.03693 | 0.00565 |
| $1022 = \sum_{i=1}^9 2^i$ | 0.02041 | 0.02038 | 0.00282 |
| $1023 = \sum_{i=0}^9 2^i$ | 0.01117 | 0.01115 | 0.00141 |
| 1024 | 0 | 0 | 0 |

Our approximation — next steps. Then we proceed with other $q - 1$ similar steps, but in each one adjusting the corresponding PDF. In each iteration, the new PDF vector becomes 0 in all positions with index smaller or equal to the index of the selected string. To compute the expected Shannon entropy, we could then update the PDF vector by calculating the weighted contribution from each possibility of each string having occurred in the previous iteration. To illustrate comparative results of Shannon entropy, we performed a Monte Carlo simulation to generate Table 12 for the uniform case,

and Table 13 for the quantum case. Each row corresponds to a pair (M, q) , where M is the pre-sample size and q is the number of strings therefrom that are selected to the final sample.

The columns ‘‘Simulation’’ show several empirical statistics, namely mean, standard deviation, minimum and maximum values, of Shannon entropy. Each row was produced from results of 1000 trials. Each trial consists of obtaining a pre-sample, from the given PDF, and then doing q iterations of the following: (i) compute the budget β of i.i.d. evaluations that would likely yield M distinct strings; (ii) compute the probabilities of each element being the lowest to be pre-sampled; (iii) increment the entropy counter based on the probability of the lowest element in the pre-sample; (iii) readjust the PDF to be 0 for its value and all lower values, and renormalize the PDF to sum to 1.

Table 12: Entropy approximations: Uniform, $q \geq 1$

Uniform (fidelity 0) case with $N = 1024$

| M | q | Simulation (1000 trials) | | | (19) | (20) |
|-----|-----|--------------------------|-------------------|---------|---------|----------|
| | | max | mean \pm stdev | min | direct | iterated |
| 1 | 1 | 10.00 | 10.00 \pm 0.0 | 10.00 | 10.00 | 10.00 |
| 2 | 2 | 19.00 | 19.00 \pm 0.0 | 18.96 | 19.00 | 19.00 |
| 4 | 2 | 26.21 | 18.09 \pm 1.3 | 16.44 | 17.00 | 16.41 |
| 4 | 4 | 35.41 | 35.41 \pm 0.0 | 35.38 | 35.42 | 35.41 |
| 8 | 2 | 24.63 | 16.53 \pm 1.6 | 14.25 | 15.00 | 14.19 |
| 8 | 4 | 42.80 | 33.03 \pm 1.8 | 29.59 | 31.42 | 29.28 |
| 8 | 8 | 64.67 | 64.66 \pm 0.0 | 64.40 | 64.70 | 64.66 |
| 32 | 2 | 22.54 | 12.89 \pm 2.0 | 10.04 | 11.00 | 10.04 |
| 32 | 8 | 65.37 | 50.92 \pm 3.5 | 43.95 | 48.70 | 41.33 |
| 32 | 24 | 166.51 | 152.90 \pm 3.4 | 144.44 | 151.00 | 137.24 |
| 32 | 32 | 201.66 | 201.63 \pm 0.0 | 201.22 | 202.34 | 201.63 |
| 512 | 8 | 36.24 | 16.16 \pm 3.9 | 8.04 | 16.70 | 8.04 |
| 512 | 32 | 95.65 | 64.26 \pm 8.0 | 45.24 | 74.34 | 32.72 |
| 512 | 256 | 566.43 | 511.07 \pm 16.3 | 463.93 | 620.00 | 318.46 |
| 512 | 512 | 1019.33 | 1018.78 \pm 0.2 | 1017.24 | 1244.83 | 1018.67 |

Legend: \pm stdev (abbreviation for standard deviation)

Table 13: Entropy approximations: Quantum, $q \geq 1$

Quantum (fidelity 1) case with $N = 1024$

| M | q | Simulation (1000 trials) | | | (19) | (20) |
|-----|-----|--------------------------|-------------------|--------|--------|----------|
| | | max | mean \pm stdev | min | direct | iterated |
| 1 | 1 | 14.67 | 9.37 \pm 1.1 | 7.05 | 9.39 | 9.39 |
| 2 | 2 | 24.78 | 17.75 \pm 1.7 | 13.60 | 17.78 | 17.78 |
| 4 | 2 | 26.06 | 16.93 \pm 2.0 | 12.42 | 15.78 | 15.19 |
| 4 | 4 | 41.39 | 32.90 \pm 2.3 | 26.36 | 32.98 | 32.96 |
| 8 | 2 | 26.44 | 15.25 \pm 2.3 | 9.87 | 13.78 | 12.97 |
| 8 | 4 | 43.99 | 30.38 \pm 2.8 | 23.00 | 28.98 | 26.83 |
| 8 | 8 | 71.03 | 59.58 \pm 3.2 | 49.45 | 59.82 | 59.76 |
| 32 | 2 | 27.53 | 11.42 \pm 2.5 | 6.07 | 9.78 | 8.82 |
| 32 | 8 | 62.02 | 46.08 \pm 4.8 | 33.76 | 43.82 | 36.43 |
| 32 | 24 | 161.98 | 138.04 \pm 6.7 | 121.30 | 136.36 | 122.40 |
| 32 | 32 | 202.65 | 182.55 \pm 6.3 | 164.42 | 182.82 | 181.73 |
| 512 | 8 | 27.96 | 11.45 \pm 4.2 | 2.21 | 11.82 | 3.14 |
| 512 | 32 | 78.16 | 46.32 \pm 8.1 | 21.25 | 54.82 | 12.83 |
| 512 | 256 | 443.12 | 368.15 \pm 20.2 | 304.75 | 463.86 | 131.37 |
| 512 | 512 | 806.84 | 734.16 \pm 21.4 | 675.63 | 932.54 | 525.15 |

The described simulation is intended only as a rough approximation. This allows us to check a degree of closeness to the results provided by formula (19). The iterated application of the formula (one string at a time) yields an entropy estimate

lower than with the direct application (using a generic q), but it is still not a lower bound for the expected entropy. In the simulated results, the rightmost column is a lower bound of the empirically observed entropy.

E. Tables with more detail

Table 14 shows sample sizes needed for several levels of FN=FP ratio in the SQC setting where the client verifies all QC-values. Table 15 shows the expected number of truncated QC-values to verify by a client, for several verification proportion budgets, when a server can pre-compute a corresponding larger amount. Table 16 shows statistics measured per bin, for various sampling budgets.

Table 14: Sample size for SQC distinguishability

| ϕ_1 | ϵ | Sample size m (number of strings) | | | | | |
|-----------|------------|-------------------------------------|---|---|---------------------------------------|---------------------------------------|---------------------------------------|
| | | when $\phi_2 = 0$ | when $\frac{\phi_2}{\phi_1} = 10^{-2}$ | when $\frac{\phi_2}{\phi_1} = 10^{-1}$ | when $\frac{\phi_2}{\phi_1} = 1/4$ | when $\frac{\phi_2}{\phi_1} = 1/2$ | when $\frac{\phi_2}{\phi_1} = 3/4$ |
| 0.002 | 2^{-40} | 4.977E+7 | 5.078E+7 | 6.146E+7 | 8.852E+7 | 1.993E+8 | 7.975E+8 |
| | 2^{-30} | 3.618E+7 | 3.692E+7 | 4.468E+7 | 6.436E+7 | 1.449E+8 | 5.798E+8 |
| | 2^{-20} | 2.273E+7 | 2.319E+7 | 2.807E+7 | 4.043E+7 | 9.102E+7 | 3.642E+8 |
| | 10^{-3} | 9.569E+6 | 9.763E+6 | 1.182E+7 | 1.702E+7 | 3.831E+7 | 1.533E+8 |
| | 10^{-2} | 5.423E+6 | 5.533E+6 | 6.696E+6 | 9.645E+6 | 2.171E+7 | 8.689E+7 |
| | 10^{-1} | 1.646E+6 | 1.679E+6 | 2.032E+6 | 2.927E+6 | 6.589E+6 | 2.637E+7 |
| | $1/4$ | 4.558E+5 | 4.651E+5 | 5.629E+5 | 8.108E+5 | 1.825E+6 | 7.304E+6 |
| | 0.01 | 2^{-40} | 2.007E+6 | 2.047E+6 | 2.480E+6 | 3.576E+6 | 8.066E+6 |
| 2^{-30} | | 1.459E+6 | 1.489E+6 | 1.803E+6 | 2.600E+6 | 5.864E+6 | 2.352E+7 |
| 2^{-20} | | 9.165E+5 | 9.352E+5 | 1.133E+6 | 1.633E+6 | 3.684E+6 | 1.477E+7 |
| 10^{-3} | | 3.858E+5 | 3.936E+5 | 4.767E+5 | 6.875E+5 | 1.551E+6 | 6.219E+6 |
| 10^{-2} | | 2.186E+5 | 2.231E+5 | 2.702E+5 | 3.896E+5 | 8.789E+5 | 3.524E+6 |
| 10^{-1} | | 6.635E+4 | 6.770E+4 | 8.199E+4 | 1.182E+5 | 2.667E+5 | 1.069E+6 |
| $1/4$ | | 1.838E+4 | 1.875E+4 | 2.271E+4 | 3.275E+4 | 7.388E+4 | 2.962E+5 |
| 0.05 | | 2^{-40} | 8.330E+4 | 8.504E+4 | 1.033E+5 | 1.499E+5 | 3.414E+5 |
| | 2^{-30} | 6.056E+4 | 6.182E+4 | 7.514E+4 | 1.090E+5 | 2.482E+5 | 1.005E+6 |
| | 2^{-20} | 3.805E+4 | 3.884E+4 | 4.720E+4 | 6.847E+4 | 1.559E+5 | 6.313E+5 |
| | 10^{-3} | 1.602E+4 | 1.635E+4 | 1.987E+4 | 2.882E+4 | 6.564E+4 | 2.657E+5 |
| | 10^{-2} | 9.077E+3 | 9.266E+3 | 1.126E+4 | 1.633E+4 | 3.720E+4 | 1.506E+5 |
| | 10^{-1} | 2.755E+3 | 2.812E+3 | 3.418E+3 | 4.957E+3 | 1.129E+4 | 4.570E+4 |
| | $1/4$ | 7.630E+2 | 7.790E+2 | 9.470E+2 | 1.374E+3 | 3.127E+3 | 1.266E+4 |
| | 0.2 | 2^{-40} | 5.827E+3 | 5.957E+3 | 7.327E+3 | 1.084E+4 | 2.551E+4 |
| 2^{-30} | | 4.237E+3 | 4.331E+3 | 5.328E+3 | 7.883E+3 | 1.855E+4 | 7.749E+4 |
| 2^{-20} | | 2.662E+3 | 2.721E+3 | 3.347E+3 | 4.953E+3 | 1.165E+4 | 4.868E+4 |
| 10^{-3} | | 1.121E+3 | 1.146E+3 | 1.409E+3 | 2.085E+3 | 4.905E+3 | 2.049E+4 |
| 10^{-2} | | 6.350E+2 | 6.490E+2 | 7.990E+2 | 1.182E+3 | 2.780E+3 | 1.161E+4 |
| 10^{-1} | | 1.930E+2 | 1.970E+2 | 2.430E+2 | 3.590E+2 | 8.440E+2 | 3.525E+3 |
| $1/4$ | | 5.400E+1 | 5.500E+1 | 6.800E+1 | 1.000E+2 | 2.340E+2 | 9.770E+2 |
| 1.0 | | 2^{-40} | 2.900E+2 | 2.980E+2 | 3.880E+2 | 6.270E+2 | 1.688E+3 |
| | 2^{-30} | 2.110E+2 | 2.170E+2 | 2.820E+2 | 4.560E+2 | 1.227E+3 | 5.786E+3 |
| | 2^{-20} | 1.330E+2 | 1.370E+2 | 1.780E+2 | 2.870E+2 | 7.710E+2 | 3.635E+3 |
| | 10^{-3} | 5.600E+1 | 5.800E+1 | 7.500E+1 | 1.210E+2 | 3.250E+2 | 1.530E+3 |
| | 10^{-2} | 3.200E+1 | 3.300E+1 | 4.300E+1 | 6.900E+1 | 1.840E+2 | 8.670E+2 |
| | 10^{-1} | 1.000E+1 | 1.000E+1 | 1.300E+1 | 2.100E+1 | 5.600E+1 | 2.640E+2 |
| | $1/4$ | 3.000E+0 | 3.000E+0 | 4.000E+0 | 6.000E+0 | 1.600E+1 | 7.300E+1 |

Legend: ϵ (FN = FP); ϕ_1 (fidelity in the honest case X_{F,m,ϕ_1}); ϕ_2 (pseudo-fidelity in the malicious case $X_{c,m,\phi_2,m}$, i.e., proportion of quantum circuit evaluations, compared to the total number m of sampled strings); FN (false negative ratio for honest case, i.e., probability of rejection when sampling with fidelity ϕ_1); FP (false positive ratio for malicious case, i.e., probability of acceptance when quantum sampling $q = \phi_2 \cdot m$ strings, and uniformly sampling $m - q$ strings); SQC (sum of **QC**-values). Values computed with the CLT approximation may yield some inaccuracies.

Table 15: Sample size for STQC distinguishability

| ϕ_1 | ϵ | Expected number $m \cdot \nu$ of TQC-values to be verified by the client | | | | | | | | |
|----------|------------|--|--|------------------------------------|----------------------|--|------------------------------------|----------------------|--|------------------------------------|
| | | when $\nu = 10^{-1}$ | | | when $\nu = 10^{-2}$ | | | when $\nu = 10^{-3}$ | | |
| | | when $\phi_2 = 0$ | when $\frac{\phi_2}{\phi_1} = 10^{-1}$ | when $\frac{\phi_2}{\phi_1} = 1/2$ | when $\phi_2 = 0$ | when $\frac{\phi_2}{\phi_1} = 10^{-1}$ | when $\frac{\phi_2}{\phi_1} = 1/2$ | when $\phi_2 = 0$ | when $\frac{\phi_2}{\phi_1} = 10^{-1}$ | when $\frac{\phi_2}{\phi_1} = 1/2$ |
| 0.002 | 2^{-40} | 7.289E+6 | 9.000E+6 | 2.918E+7 | 2.241E+6 | 2.768E+6 | 8.986E+6 | 1.036E+6 | 1.280E+6 | 4.160E+6 |
| | 2^{-30} | 5.299E+6 | 6.543E+6 | 2.122E+7 | 1.629E+6 | 2.013E+6 | 6.533E+6 | 7.535E+5 | 9.309E+5 | 3.025E+6 |
| | 2^{-20} | 3.329E+6 | 4.111E+6 | 1.333E+7 | 1.024E+6 | 1.264E+6 | 4.104E+6 | 4.734E+5 | 5.848E+5 | 1.900E+6 |
| | 10^{-3} | 1.401E+6 | 1.730E+6 | 5.611E+6 | 4.309E+5 | 5.322E+5 | 1.728E+6 | 1.993E+5 | 2.462E+5 | 7.998E+5 |
| | 10^{-2} | 7.941E+5 | 9.806E+5 | 3.180E+6 | 2.442E+5 | 3.016E+5 | 9.791E+5 | 1.129E+5 | 1.395E+5 | 4.533E+5 |
| | 10^{-1} | 2.410E+5 | 2.976E+5 | 9.650E+5 | 7.411E+4 | 9.154E+4 | 2.971E+5 | 3.427E+4 | 4.234E+4 | 1.376E+5 |
| | 1/4 | 6.676E+4 | 8.243E+4 | 2.673E+5 | 2.053E+4 | 2.536E+4 | 8.231E+4 | 9.493E+3 | 1.173E+4 | 3.811E+4 |
| 0.01 | 2^{-40} | 2.967E+5 | 3.667E+5 | 1.193E+6 | 9.343E+4 | 1.156E+5 | 3.781E+5 | 4.432E+4 | 5.490E+4 | 1.804E+5 |
| | 2^{-30} | 2.157E+5 | 2.666E+5 | 8.675E+5 | 6.793E+4 | 8.406E+4 | 2.749E+5 | 3.222E+4 | 3.992E+4 | 1.311E+5 |
| | 2^{-20} | 1.355E+5 | 1.675E+5 | 5.450E+5 | 4.267E+4 | 5.281E+4 | 1.727E+5 | 2.024E+4 | 2.508E+4 | 8.239E+4 |
| | 10^{-3} | 5.705E+4 | 7.050E+4 | 2.294E+5 | 1.796E+4 | 2.223E+4 | 7.270E+4 | 8.520E+3 | 1.056E+4 | 3.468E+4 |
| | 10^{-2} | 3.233E+4 | 3.996E+4 | 1.300E+5 | 1.018E+4 | 1.260E+4 | 4.120E+4 | 4.829E+3 | 5.983E+3 | 1.965E+4 |
| | 10^{-1} | 9.811E+3 | 1.213E+4 | 3.945E+4 | 3.090E+3 | 3.823E+3 | 1.250E+4 | 1.466E+3 | 1.816E+3 | 5.965E+3 |
| | 1/4 | 2.718E+3 | 3.359E+3 | 1.093E+4 | 8.560E+2 | 1.059E+3 | 3.464E+3 | 4.060E+2 | 5.030E+2 | 1.653E+3 |
| 0.05 | 2^{-40} | 1.295E+4 | 1.607E+4 | 5.318E+4 | 4.549E+3 | 5.681E+3 | 1.924E+4 | 2.403E+3 | 3.017E+3 | 1.042E+4 |
| | 2^{-30} | 9.413E+3 | 1.168E+4 | 3.867E+4 | 3.307E+3 | 4.131E+3 | 1.399E+4 | 1.747E+3 | 2.194E+3 | 7.572E+3 |
| | 2^{-20} | 5.913E+3 | 7.340E+3 | 2.429E+4 | 2.078E+3 | 2.595E+3 | 8.789E+3 | 1.098E+3 | 1.378E+3 | 4.757E+3 |
| | 10^{-3} | 2.489E+3 | 3.090E+3 | 1.023E+4 | 8.750E+2 | 1.093E+3 | 3.700E+3 | 4.620E+2 | 5.810E+2 | 2.003E+3 |
| | 10^{-2} | 1.411E+3 | 1.751E+3 | 5.795E+3 | 4.960E+2 | 6.190E+2 | 2.097E+3 | 2.620E+2 | 3.290E+2 | 1.135E+3 |
| | 10^{-1} | 4.290E+2 | 5.320E+2 | 1.759E+3 | 1.510E+2 | 1.880E+2 | 6.370E+2 | 8.000E+1 | 1.000E+2 | 3.450E+2 |
| | 1/4 | 1.190E+2 | 1.480E+2 | 4.880E+2 | 4.200E+1 | 5.300E+1 | 1.770E+2 | 2.300E+1 | 2.800E+1 | 9.600E+1 |
| 0.2 | 2^{-40} | 1.105E+3 | 1.396E+3 | 4.913E+3 | 5.260E+2 | 6.770E+2 | 2.542E+3 | 3.500E+2 | 4.580E+2 | 1.789E+3 |
| | 2^{-30} | 8.040E+2 | 1.015E+3 | 3.572E+3 | 3.820E+2 | 4.930E+2 | 1.848E+3 | 2.540E+2 | 3.330E+2 | 1.301E+3 |
| | 2^{-20} | 5.050E+2 | 6.380E+2 | 2.244E+3 | 2.400E+2 | 3.100E+2 | 1.161E+3 | 1.600E+2 | 2.090E+2 | 8.170E+2 |
| | 10^{-3} | 2.130E+2 | 2.690E+2 | 9.450E+2 | 1.010E+2 | 1.310E+2 | 4.890E+2 | 6.800E+1 | 8.800E+1 | 3.440E+2 |
| | 10^{-2} | 1.210E+2 | 1.530E+2 | 5.360E+2 | 5.800E+1 | 7.400E+1 | 2.770E+2 | 3.900E+1 | 5.000E+1 | 1.950E+2 |
| | 10^{-1} | 3.700E+1 | 4.700E+1 | 1.630E+2 | 1.800E+1 | 2.300E+1 | 8.500E+1 | 1.200E+1 | 1.600E+1 | 6.000E+1 |
| | 1/4 | 1.100E+1 | 1.300E+1 | 4.500E+1 | 5.000E+0 | 7.000E+0 | 2.400E+1 | 4.000E+0 | 5.000E+0 | 1.700E+1 |

Legend: ϵ (FN = FP); ϕ_1 (fidelity in the honest case Y_{F,m,ϕ_1}); ϕ_2 (pseudo-fidelity in the malicious case $Y_{E,m,\phi_2,m}$; i.e., proportion of quantum circuit evaluations, compared to the total number m of sampled strings); FN (**f**alse **n**egative rate for honest case, i.e., probability of rejection when sampling with fidelity ϕ_1); FP (**f**alse **p**ositive rate for malicious case, i.e., probability of acceptance when quantum sampling $q = \phi_2 \cdot m$ strings, and uniformly sampling $m - q$ strings); STQC (sum of truncated **Q**C-values). Values computed with the CLT approximation may yield some inaccuracies.

Table 16: Statistics per bin c and budget factor b

Values obtained from direct measurement of (102), (103) and (105), for a discretized PDF for 28 qubits.

| | $b = \beta/N$ | 2^{-21} | 2^{-17} | 2^{-13} | 2^{-9} | 2^{-5} | 2^{-2} | 2^{-1} | 2^0 | 2^1 | 2^2 | 2^4 |
|-----------|---------------|--------------|--------------|--------------|--------------|--------------|---------------|---------------|---------------|---------------|---------------|---------------|
| $c = 0$ | $\log_2(M/N)$ | -0.000 | -0.000 | -0.000 | -0.003 | -0.044 | -0.322 | -0.585 | -1.000 | -1.585 | -2.322 | -4.087 |
| | $N \cdot A$ | 1.000 | 1.000 | 1.000 | 0.998 | 0.970 | 0.800 | 0.667 | 0.500 | 0.333 | 0.200 | 0.059 |
| | $n - h$ | -0.000 | 0.000 | 0.000 | 0.000 | 0.001 | 0.033 | 0.104 | 0.279 | 0.623 | 1.168 | 2.730 |
| $c = 1$ | $\log_2(M/N)$ | -21.000 | -17.000 | -13.000 | -9.006 | -5.089 | -2.644 | -2.170 | -2.000 | -2.170 | -2.644 | -4.175 |
| | $N \cdot A$ | 2.000 | 2.000 | 2.000 | 1.996 | 1.939 | 1.600 | 1.333 | 1.000 | 0.667 | 0.400 | 0.118 |
| | $n - h$ | 0.610 | 0.610 | 0.610 | 0.607 | 0.567 | 0.355 | 0.233 | 0.167 | 0.271 | 0.624 | 1.982 |
| $c = 2$ | $\log_2(M/N)$ | -42.011 | -34.001 | -26.001 | -18.008 | -10.133 | -4.966 | -3.755 | -3.000 | -2.755 | -2.966 | -4.262 |
| | $N \cdot A$ | 3.000 | 3.000 | 3.000 | 2.994 | 2.909 | 2.400 | 2.000 | 1.500 | 1.000 | 0.600 | 0.176 |
| | $n - h$ | <u>1.663</u> | <u>1.663</u> | 1.662 | 1.657 | 1.576 | 1.119 | 0.805 | 0.499 | 0.362 | 0.522 | 1.677 |
| $c = 3$ | $\log_2(M/N)$ | — | -51.002 | -39.001 | -27.011 | -15.178 | -7.288 | -5.340 | -4.000 | -3.340 | -3.288 | -4.350 |
| | $N \cdot A$ | — | 4.000 | 4.000 | 3.992 | 3.879 | 3.200 | 2.667 | 2.000 | 1.333 | 0.800 | 0.235 |
| | $n - h$ | — | <u>2.852</u> | <u>2.851</u> | 2.843 | 2.721 | 2.019 | 1.513 | 0.966 | 0.589 | 0.557 | 1.508 |
| $c = 4$ | $\log_2(M/N)$ | — | — | -52.001 | -36.014 | -20.222 | -9.610 | -6.925 | -5.000 | -3.925 | -3.610 | -4.437 |
| | $N \cdot A$ | — | — | 4.999 | 4.990 | 4.848 | 4.000 | 3.333 | 2.500 | 1.667 | 1.000 | 0.294 |
| | $n - h$ | — | — | <u>4.106</u> | <u>4.095</u> | 3.932 | 2.986 | 2.287 | 1.500 | 0.882 | 0.658 | 1.405 |
| $c = 5$ | $\log_2(M/N)$ | — | — | — | -45.017 | -25.266 | -11.932 | -8.510 | -6.000 | -4.510 | -3.932 | -4.525 |
| | $N \cdot A$ | — | — | — | 5.988 | 5.818 | 4.800 | 4.000 | 3.000 | 2.000 | 1.200 | 0.353 |
| | $n - h$ | — | — | — | <u>5.386</u> | 5.182 | 3.991 | 3.100 | 2.072 | 1.214 | 0.797 | 1.341 |
| $c = 6$ | $\log_2(M/N)$ | — | — | — | — | -30.311 | -14.253 | -10.095 | -7.000 | -5.095 | -4.253 | -4.612 |
| | $N \cdot A$ | — | — | — | — | 6.787 | 5.600 | 4.667 | 3.500 | 2.333 | 1.400 | 0.412 |
| | $n - h$ | — | — | — | — | <u>6.457</u> | 5.021 | 3.938 | 2.670 | 1.572 | 0.962 | 1.302 |
| $c = 8$ | $\log_2(M/N)$ | — | — | — | — | -40.400 | -18.897 | -13.265 | -9.000 | -6.265 | -4.897 | -4.787 |
| | $N \cdot A$ | — | — | — | — | 8.722 | 7.200 | 6.000 | 4.500 | 3.000 | 1.800 | 0.529 |
| | $n - h$ | — | — | — | — | <u>9.055</u> | 7.132 | 5.664 | 3.915 | 2.336 | 1.342 | 1.274 |
| $c = 16$ | $\log_2(M/N)$ | — | — | — | — | — | -37.492 | -25.945 | -17.000 | -10.944 | -7.473 | -5.487 |
| | $N \cdot A$ | — | — | — | — | — | 13.453 | 11.325 | 8.500 | 5.667 | 3.400 | 1.000 |
| | $n - h$ | — | — | — | — | — | <u>15.759</u> | 12.869 | 9.201 | 5.698 | 3.165 | 1.468 |
| $c = 32$ | $\log_2(M/N)$ | — | — | — | — | — | — | -52.413 | -33.057 | -20.304 | -12.624 | -6.886 |
| | $N \cdot A$ | — | — | — | — | — | — | 18.609 | 16.185 | 11.000 | 6.600 | 1.941 |
| | $n - h$ | — | — | — | — | — | — | <u>24.068</u> | <u>20.004</u> | 12.900 | 7.289 | 2.333 |
| $c = 64$ | $\log_2(M/N)$ | — | — | — | — | — | — | — | — | -40.088 | -22.925 | -9.685 |
| | $N \cdot A$ | — | — | — | — | — | — | — | — | 19.251 | 13.000 | 3.824 |
| | $n - h$ | — | — | — | — | — | — | — | — | <u>25.418</u> | 16.026 | 4.553 |
| $c = 128$ | $\log_2(M/N)$ | — | — | — | — | — | — | — | — | — | -49.796 | -15.283 |
| | $N \cdot A$ | — | — | — | — | — | — | — | — | — | 20.032 | 7.588 |
| | $n - h$ | — | — | — | — | — | — | — | — | — | <u>27.500</u> | 9.486 |
| $c = 256$ | $\log_2(M/N)$ | — | — | — | — | — | — | — | — | — | — | -26.478 |
| | $N \cdot A$ | — | — | — | — | — | — | — | — | — | — | 15.118 |
| | $n - h$ | — | — | — | — | — | — | — | — | — | — | 19.850 |
| $c = 384$ | $\log_2(M/N)$ | — | — | — | — | — | — | — | — | — | — | -41.657 |
| | $N \cdot A$ | — | — | — | — | — | — | — | — | — | — | 20.112 |
| | $n - h$ | — | — | — | — | — | — | — | — | — | — | <u>27.852</u> |

Legend: A (expected QC-value of strings in bin c); b (budget factor = β/N); β (sampling budget — number of quantumly-sampled strings); c (multiplicity); h (average entropy per string in bin c); M (expected number of strings in bin c); n (number of qubits); N (string space size); ϕ (fidelity). Note: the values were obtained from the probability distributions for $n = 28$ qubits and $\phi = 1$. Since we are actually interested in the case of 53 qubits, we redacted (—) the entries corresponding to $\log_2(M/N) \leq 2^{-53}$. The entries for $n - h$ are underlined when $2^{-53} < \log_2(M/N) < 2^{-28}$, case in which the results may be inaccurate for the case of 53 qubits. The entries for $\log_2(M/N)$ and $N \cdot A$ show underlined the digits that differ from the result that would be obtained with formulas (112) for M and (113) for A .