

Noisy Simon Period Finding

Alexander May* , Lars Schlieper* , and Jonathan Schwinger

Horst Görtz Institute for IT Security
 Ruhr-University Bochum, Germany
 {alex.may,lars.schlieper,jonathan.schwinger}@rub.de

Abstract. Let $f : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n$ be a Boolean function with period \mathbf{s} . It is well-known that Simon’s algorithm finds \mathbf{s} in time polynomial in n on quantum devices that are capable of performing error-correction. However, today’s quantum devices are inherently noisy, too limited for error correction, and Simon’s algorithm is not error-tolerant.

We show that even noisy quantum period finding computations may lead to speedups in comparison to purely classical computations. To this end, we implemented Simon’s quantum period finding circuit on the 15-qubit quantum device IBM Q 16 Melbourne. Our experiments show that with a certain probability $\tau(n)$ we measure erroneous vectors that are not orthogonal to \mathbf{s} . We propose new, simple, but very effective smoothing techniques to classically mitigate physical noise effects such as e.g. IBM Q’s bias towards the 0-qubit.

After smoothing, our noisy quantum device provides us a statistical distribution that we can easily transform into an LPN instance with parameters n and $\tau(n)$. Hence, in the noisy case we may not hope to find periods in time polynomial in n . However, we may still obtain a quantum advantage if the error $\tau(n)$ does not grow too large. This demonstrates that quantum devices may be useful for period finding, even before achieving the level of full error correction capability.

Keywords: Noise-tolerant Simon period finding, IBM-Q16, LPN, quantum advantage

1 Introduction

The discovery of Shor’s quantum algorithm [23] for factoring and computing discrete logarithms in 1994 had a dramatic impact on public-key cryptography, initiating the fast growing field of post-quantum cryptography that studies problems supposed to be hard even on quantum computers, such as e.g. Learning Parity with Noise (LPN) [3] and Learning with Errors (LWE) [20].

For some decades, the common belief was that the impact of quantum algorithms on symmetric crypto is way less dramatic, since the effect of Grover search can easily be handled by doubling the key size. However, starting with

* Funded by DFG under Germany’s Excellence Strategy - EXC 2092 CASA - 390781972.

the initial work of Kuwakado, Morii [17] and followed by Kaplan, Leurent, Leverrier and Naya-Plasencia [15] it was shown that (among others) the well-known Even-Mansour construction can be broken with quantum CPA-attacks [5] in polynomial time using Simon’s quantum period finding algorithm [24]. This is especially interesting, because Even and Mansour [12] proved that in the ideal cipher model any classical attack on their construction with n -bit keys requires $\Omega(2^{\frac{n}{2}})$ steps.

These results triggered a whole line of work that studies the impact of Simon’s algorithm and its variants for symmetric key cryptography, including e.g. [21,18,2,6,14,8,7]. In a nutshell, Simon’s quantum circuit produces for a periodic function $f : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n$ with period $\mathbf{s} \in \mathbb{F}_2^n$, i.e. $f(\mathbf{x}) = f(\mathbf{z})$ iff $\mathbf{z} \in \{\mathbf{x}, \mathbf{x} + \mathbf{s}\}$, via quantum measurements uniformly distributed vectors \mathbf{y} that are orthogonal to \mathbf{s} . It is not hard to see that from a basis of \mathbf{y} ’s that spans the subspace orthogonal to \mathbf{s} , the period \mathbf{s} can be computed via elementary linear algebra in time polynomial in n . Thus, Simon’s algorithm finds the period with a linear number of quantum measurements (and calls to f), and some polynomial time classical post-processing. On any purely classical computer however, finding the period of f requires in general $\Omega(2^{\frac{n}{2}})$ operations [19]. Let us stress again that we consider quantum CPA attacks via Simon, i.e. the attacker has access to a cipher that is implemented quantumly—a very powerful attack model.

Our contributions. We implemented Simon’s algorithm on IBM’s freely available Q16 MELBOURNE [1], called IBM-Q16 in the following, that realizes 15-qubit quantum circuits. Since Simon’s quantum circuit requires for n -bit periodic functions $2n$ qubits, we were able to implement functions up to $n = 7$ bits. Due to its limited size, IBM-Q16 is not capable of performing full error correction [9] for $n > 1$. However, we show that error correction is no necessary requirement for achieving quantum speedups.

Implementation. Our experiments show that with some (significant) error probability τ , we measure on IBM-Q16 vectors \mathbf{y} that are *not orthogonal* to \mathbf{s} . The error probability τ depends on many factors, such as the number of 1- and 2-qubit gates that we use to realize Simon’s circuit, IBM-Q16’s topology that allows only limited 2-qubit applications, and even the individual qubits that we use. We optimize our Simon implementation to achieve minimal error τ . Since increasing n requires an increasing amount of gates, we discover experimentally that $\tau(n)$ grows as a function of n . For the function f that we implemented, we found τ -values ranging between $\tau(2) = 0.09$ and $\tau(7) = 0.13$. We would like to stress that our choice of f is highly optimized to minimize IBM-Q16’s error. Any realistic real-word cryptographic f would at the moment result in outputs close to random noise, i.e. with $\tau(n)$ close to $\frac{1}{2}$.

For our simple f despite the errors we still qualitatively observe the desired quantum effect: Vectors \mathbf{y} orthogonal to \mathbf{s} appear with significant larger probabilities than vectors not orthogonal to \mathbf{s} . Similar experimental observations have been achieved in Tame et al. [25].

Smoothing techniques. In the error free case, Simon’s circuit produces vectors that are uniformly distributed. However, on IBM-Q16 this is not the case. First, IBM-Q16’s qubits have different noise level, hence different reliability. Second, we experimentally observe vectors with small Hamming weight more frequently, the measured qubits have a bias towards 0.

To mitigate both effects we introduce simple, but effective *smoothing techniques*. First, the quality of qubits can be averaged by introducing permutations that preserve the overall error probability τ . Second, the 0-bias can be removed by suitable addition of vectors, both quantumly and classically. In combination, our smoothing methods are effective in the sense that they provide a distribution where vectors orthogonal to \mathbf{s} appear uniformly distributed with probability $1 - \tau$, and vectors not orthogonal to \mathbf{s} appear uniformly distributed with probability τ . Note that our smoothing techniques do not reduce the overall error τ , but smooth the error distribution.

We call the problem of recovering $\mathbf{s} \in \mathbb{F}_2^n$ from such a distribution *Learning Simon with Noise* (LSN) with parameters n and τ . Notice that intuitively it should be hard to distinguish orthogonal vectors from non-orthogonal ones.

Hardness. We show that solving LSN with parameters n, τ is tightly polynomial time equivalent to solving the famous *Learning Parity with Noise* (LPN) problem with the same parameters n, τ . The core of our reduction shows that LSN samples coming from smoothed quantum measurements of Simon’s circuit can be turned into perfectly distributed LPN samples, and vice versa. Hence, smoothed quantum measurements of Simon’s circuit realize a *physical LPN oracle*.

From an error-tolerance perspective, our LPN-to-LSN reduction may at first sound quite negative, since it is common belief that we cannot solve LPN (and thus also not LSN) in time polynomial in (n, τ) — not even on a quantum computer.

Error Handling. On the positive side, we may use the converse LSN-to-LPN reduction to handle errors from noisy quantum devices like IBM-Q16 via LPN-solving algorithms. Theoretically, the best algorithm for solving LPN with constant τ is the BKW-algorithm of Blum, Kalai and Wasserman [4] with time complexity $2^{\mathcal{O}\left(\frac{n}{\log(\frac{n}{\tau})}\right)}$. This already improves on the classical time $2^{\frac{n}{2}}$ for period finding.

Practically, the current LPN records with errors $\tau \in [0.09, 0.13]$ —as observed in our IBM-Q16 experiments—are solved with variants of the algorithms POOLED GAUSS and WELL-POOLED GAUSS of Esser, Kübler, May [11]. We show that POOLED GAUSS solves LSN for $\tau \leq 0.292$ faster than classical period finding algorithms. WELL-POOLED GAUSS even improves on any classical period finding algorithm for all errors $\tau < \frac{1}{2}$.

WELL-POOLED GAUSS is able to handle errors in time 2^{cn} , where $c < \frac{1}{2}$ is constant for constant τ . For the error-free case $\tau = 0$, we obtain polynomial time as predicted by Simon’s analysis. In the noisy case $0 < \tau < \frac{1}{2}$ we achieve

exponential run time, yet still improve over purely classical computation. This indicates that we achieve *quantum advantage* for the Simon period finding problem on sufficiently large computers, even in the presence of errors: Our quantum oracle helps us in speeding up computation! But as opposed to the exponential speedup from the (unrealistic) error-free Simon setting $\tau = 0$, we obtain in the practically relevant noisy Simon setting $0 < \tau < \frac{1}{2}$ only a *polynomial speedup* with a polynomial of degree $\frac{1}{2c} > 1$.

Assume that in a possibly far future one could build a quantum device with 486 qubits performing Simon’s circuit on a 243-bit *realistic real-world cryptographic* periodic function with error $\tau(486) = \frac{1}{8}$. Then our smoothed techniques could translate the noisy quantum data into an LPN-instance with $(n, \tau) = (243, \frac{1}{8})$. Such an LPN instance was solved in [11] on 64 threads in only 15 days, whereas classically period finding would require 2^{121} steps.

We would like to stress that our introduction of a simple error parameter τ is to indicate at which point in the future quantum devices may help to speed up Simon-based quantum cryptanalysis. We do not give any predictions how $\tau(n)$ behaves for future devices, nor for realistic cryptographic functions. This remains an open problem.

Our paper is organized as follows. In Section 2 we recall Simon’s original quantum circuit, and already introduce our LSN Error Model. In Section 3 we run IBM-Q16 experiments, and show in Section 4 how to smooth the results of the quantum computations¹ such that they fit our error model. In Section 5 we show the polynomial time equivalence of LSN and LPN. In Section 6 we theoretically show that quantum measurements with error τ in combination with LPN-solvers outperform classical period finding for any $\tau < \frac{1}{2}$. Eventually, in Section 7 we experimentally extract periods from noisy IBM-Q16 measurements.

2 Simon’s Algorithm in the Noisy Case

Notation. All logs in this paper are base 2. Let $\mathbf{x} \in \mathbb{F}_2^n$ denote a binary vector with coordinates $\mathbf{x} = (x_{n-1}, \dots, x_0)$ and Hamming weight $h(\mathbf{x}) = \sum_{i=0}^{n-1} x_i$. Let $\mathbf{0} \in \mathbb{F}_2^n$ be the vector with all-zero coordinates. We denote by \mathcal{U} the uniform distribution over \mathbb{F}_2 , and by \mathcal{U}_n the uniform distribution over \mathbb{F}_2^n . If a random variable X is chosen from distribution \mathcal{U} , we write $X \sim \mathcal{U}$. We denote by Ber_τ the Bernoulli distribution for \mathbb{F}_2 , i.e. a 0, 1-valued $X \sim \text{Ber}_\tau$ with $\mathbb{P}[X = 1] = \tau$.

Two vectors \mathbf{x}, \mathbf{y} are *orthogonal* if their inner product $\langle \mathbf{x}, \mathbf{y} \rangle := \sum_{i=0}^{n-1} x_i y_i \pmod 2$ is 0, otherwise they are called *non-orthogonal*. Let $\mathbf{s} \in \mathbb{F}_2^n$. Then we denote the subspace of all vectors orthogonal to \mathbf{s} as

$$\mathbf{s}^\perp = \{\mathbf{x} \in \mathbb{F}_2^n \mid \langle \mathbf{x}, \mathbf{s} \rangle = 0\}.$$

Let $Y = \{\mathbf{y}_1, \dots, \mathbf{y}_k\} \subseteq \mathbb{F}_2^n$. Then we define $Y^\perp = \{\mathbf{x} \mid \langle \mathbf{x}, \mathbf{y}_i \rangle = 0 \text{ for all } i\}$.

¹ IBM-Q16 data can be found in our supplementary material.

For a Boolean function $f : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n$ we denote its *universal (quantum) embedding* by

$$U_f : \mathbb{F}_2^{2n} \rightarrow \mathbb{F}_2^{2n} \text{ with } (\mathbf{x}, \mathbf{y}) \mapsto (\mathbf{x}, f(\mathbf{x}) + \mathbf{y}).$$

Notice that $U_f(U_f(\mathbf{x}, \mathbf{y})) = (\mathbf{x}, \mathbf{y})$.

Let $|x\rangle \in \mathbb{C}^2$ with $x \in \mathbb{F}_2$ be a qubit. We denote by H the *Hadamard function*

$$x \mapsto \frac{1}{\sqrt{2}}(|0\rangle + (-1)^x |1\rangle).$$

We briefly write H_n for the n -fold tensor product $H \otimes \dots \otimes H$. Let $|x\rangle |y\rangle \in \mathbb{C}^4$ be a 2-qubit system. The **cnot** (controlled **not**) function is the universal embedding of the identity function, i.e. $|x\rangle |y\rangle \mapsto |x\rangle |x + y\rangle$. We call the first qubit $|x\rangle$ *control bit*, since we perform a **not** on $|y\rangle$ iff $x = 1$.

A *Simon function* is a periodic $(2 : 1)$ -Boolean function defined as follows.

Definition 2.1 (Simon function/problem). *Let $f : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n$. We call f a Simon function if there exists some period $\mathbf{s} \in \mathbb{F}_2^n \setminus \{\mathbf{0}\}$ such that for all $\mathbf{x} \neq \mathbf{y} \in \mathbb{F}_2^n$ we have*

$$f(\mathbf{x}) = f(\mathbf{y}) \Leftrightarrow \mathbf{y} = \mathbf{x} + \mathbf{s}.$$

In Simon's problem we have to find \mathbf{s} given oracle access to f .

In order to solve Simon's problem classically, we have to find some collision $\mathbf{x} \neq \mathbf{y}$ satisfying $f(\mathbf{x}) = f(\mathbf{y})$. It is well-known that this requires $\Omega(2^{\frac{n}{2}})$ function evaluations.

Simon's quantum algorithm [24], called SIMON (see Algorithm 1), solves Simon's problem with only $\mathcal{O}(n)$ function evaluations on a quantum circuit. It is known that on input $|0^n\rangle \otimes |0^n\rangle$ a measurement of the first n qubits of the quantum circuit Q_f^{SIMON} depicted in Figure 1 yields some $\mathbf{y} \in \mathbb{F}_2^n$ that is orthogonal to \mathbf{s} . Moreover, $\mathbf{y} \in \mathbb{F}_2^n$ is uniformly distributed in the subspace \mathbf{s}^\perp , i.e. we obtain each $\mathbf{y} \in \mathbf{s}^\perp$ with probability $\frac{1}{2^{n-1}}$. SIMON repeats to measure Q_f^{SIMON} until it has collected $n - 1$ linearly independent vectors $\mathbf{y}_1, \dots, \mathbf{y}_{n-1}$, from which \mathbf{s} can be computed via linear algebra in polynomial time. It is not hard to see that the collection of $n - 1$ linearly independent vectors requires only $\mathcal{O}(n)$ function evaluations.

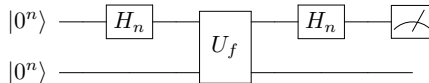


Fig. 1: Quantum circuit Q_f^{SIMON}

At this point we should stress that SIMON only works for *noiseless* quantum computations. Hence we have to ensure that each \mathbf{y} is indeed in \mathbf{s}^\perp . Assume that we obtain in line 4 of algorithm SIMON at least a single \mathbf{y} with $\langle \mathbf{y}, \mathbf{s} \rangle = 1$. Then the output of SIMON is always false! Thus, SIMON is not robust against noisy quantum computations.

More precisely, if we obtain in line 4 erroneous $\mathbf{y} \notin \mathbf{s}^\perp$ with probability τ , $0 < \tau \leq \frac{1}{2}$, then SIMON outputs the correct \mathbf{s} only with exponentially small probability success probability $(1 - \tau)^n$. This motivates our following quite simple error model.

Algorithm 1: SIMON

Input : Simon function $f : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n$.
Output: Period $\mathbf{s} \in \mathbb{F}_2^n$

- 1 Set $Y = \emptyset$.
- 2 **repeat**
- 3 Run Q_f^{SIMON} on $|0^n\rangle \otimes |0^n\rangle$.
- 4 Let $\mathbf{y} \in \mathbb{F}_2^n$ be the measurement of the first n qubits.
- 5 If $\mathbf{y} \notin \text{span}(Y)$, then include \mathbf{y} in Y .
- 6 **until** $|Y| = n - 1$
- 7 Compute the unique $\mathbf{s} \in Y^\perp \setminus \{\mathbf{0}\}$.
- 8 **return** \mathbf{s} .

Definition 2.2 (LSN Error Model). Let $\tau \in \mathbb{R}$ with $0 \leq \tau \leq \frac{1}{2}$. Upon measuring the first n qubits of Q_f^{SIMON} , our quantum device outputs with probability $1 - \tau$ some uniformly random $\mathbf{y} \in \mathbf{s}^\perp$, and with probability τ some uniformly random $\mathbf{y} \in \mathbb{F}_2^n \setminus \mathbf{s}^\perp$. That is, the output distribution is

$$\mathbb{P}[Q_f^{\text{SIMON}} \text{ outputs } \mathbf{y}] = \begin{cases} \frac{1-\tau}{2^{n-1}} & \text{if } \mathbf{y} \in \mathbf{s}^\perp \\ \frac{\tau}{2^n-1} & \text{else} \end{cases}. \quad (1)$$

We call τ the error rate of our quantum device. We call the problem of computing \mathbf{s} from the distribution in Equation (1) Learning Simon with Noise (LSN). We further refine LSN in Definition 5.2.

In the subsequent Section 3 we show that the results of our IBM-Q16 implementation only roughly follows the LSN Error Model of Definition 2.2. However, we also introduce in Section 4 simple smoothing techniques such that the IBM-Q16 measurements can be transformed into almost perfectly matching our error model.

Notice that intuitively there is no efficient way to tell whether $\mathbf{y} \in \mathbf{s}^\perp$. This intuition is confirmed in Section 5, where we show that solving LSN is tightly as hard as solving the Learning Parity with Noise (LPN) problem.

3 Quantum Period Finding on IBM-Q16

We ran our experiments on the IBM-Q16 Melbourne device, which (despite its name) realizes 15-qubit circuits. Let us number IBM-Q16's qubits as $0, \dots, 14$. Our implementation goal was to realize quantum period finding for Simon functions $f : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n$ with error rate as small as possible. To this end we used the following optimization criteria.

Gate count. IBM-Q16 realizes several 1-qubit gates such as Hadamard and rotations, but only the 2-qubit gate **cnot**. On IBM-Q16, the application of any gates introduces some error, where especially the 2-qubit **cnot** introduces

approximately as much error as ten 1-qubit gates (see Figure 2). Therefore, we introduce a circuit norm that defines a weighted gate count, which we minimize in the following.

Definition 3.1. Let Q be a quantum circuit with g_1 many 1-qubit gates and g_2 many 2-qubit gates. Then we define Q 's circuit-norm as $CN(Q) := g_1 + 10g_2$.

Topology. IBM-Q16 can only process 2-qubit gates on qubits that are adjacent in its topology graph, see Figure 2. Let $G = (V, E)$ be the undirected topology graph, where node i denotes qubit i .

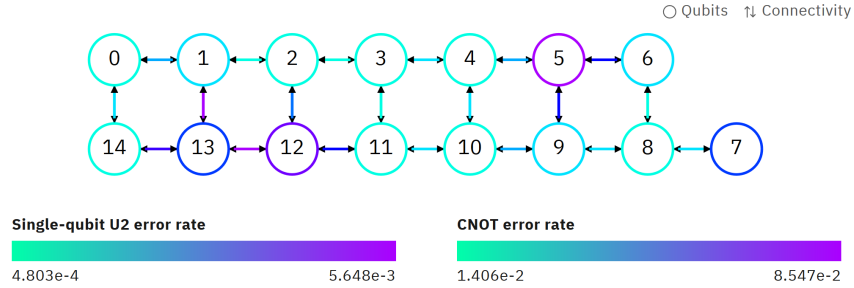


Fig. 2: Topology graph $G(V, E)$ of IBM-Q16.

If $\{u, v\} \in E$ then we can directly implement $\mathbf{cnot}(u, v)$, respectively $\mathbf{cnot}(v, u)$, where u , respectively v , serves as the control bit. Hence, we call qubits u, v *adjacent* iff $\{u, v\} \in E$.

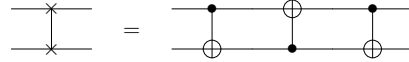


Fig. 3: Realisation of **swap** via 3 **cnots**.

Let us assume that we want to realize $\mathbf{cnot}(1, 3)$ in our algorithm. Since $\{1, 3\} \notin E$ we may first swap the contents of qubits 2 and 3 by realizing a **swap** gate via 3 **cnots** as depicted in Figure 3. Thus, with a total of 3 **cnots** we swap the content of qubit 3 into 2. Since $\{1, 2\} \in E$, we may now apply $\mathbf{cnot}(1, 2)$.

3.1 Function Choice

Notice that in Definition 2.1 of Simon's problem, we obtain oracle access to a Simon function f . In a quantum-CPA attack we assume that a cryptographic function f is realized via its quantum embedding U_f . An attacker gets black-box access to U_f , i.e. he can query U_f on inputs of his choice in superposition.

We choose the following function f_s whose U_{f_s} is not too expensive to realize on IBM-Q16.

Definition 3.2. Let $\mathbf{s} \in \mathbb{F}_2^n \setminus \{\mathbf{0}\}$, and let $i \in [0, n - 1]$ be the smallest i with $s_i = 1$. We define

$$f_{\mathbf{s}} : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n, \quad \mathbf{x} \mapsto \mathbf{x} + x_i \cdot \mathbf{s}.$$

Let us first show that $f_{\mathbf{s}}$ is indeed a Simon function as given in Definition 2.1. Moreover, we show that *every* Simon function – no matter whether it is efficiently computable or not – is of the form $f_{\mathbf{s}}$ followed by some permutation.

Lemma 3.1. Let $f_{\mathbf{s}}(\mathbf{x}) = \mathbf{x} + x_i \cdot \mathbf{s}$ as in Definition 3.2. Then the following holds.

- (1) $f_{\mathbf{s}}$ is a Simon function with period \mathbf{s} , i.e. $f_{\mathbf{s}}(\mathbf{x}) = f_{\mathbf{s}}(\mathbf{y})$ iff $\mathbf{y} \in \{\mathbf{x}, \mathbf{x} + \mathbf{s}\}$.
- (2) Any Simon function is of the form $P \circ f_{\mathbf{s}}$ for some bijection $P : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n$.

Proof. (1) We have for all $\mathbf{x} \in \mathbb{F}_2^n$ that

$$f_{\mathbf{s}}(\mathbf{x} + \mathbf{s}) = \mathbf{x} + \mathbf{s} + (\mathbf{x} + \mathbf{s})_i \cdot \mathbf{s} = \mathbf{x} + \mathbf{s} + (x_i + 1) \cdot \mathbf{s} = \mathbf{x} + x_i \cdot \mathbf{s} = f_{\mathbf{s}}(\mathbf{x}).$$

Thus, f has period \mathbf{s} . It remains to show that $f_{\mathbf{s}}$ is $(2 : 1)$, i.e. that $f_{\mathbf{s}}(\mathbf{x}) = f_{\mathbf{s}}(\mathbf{y})$ implies that $\mathbf{y} = \mathbf{x}$ or $\mathbf{y} = \mathbf{x} + \mathbf{s}$. From $f_{\mathbf{s}}(\mathbf{x}) = f_{\mathbf{s}}(\mathbf{y})$ we conclude

$$\mathbf{x} + x_i \cdot \mathbf{s} = \mathbf{y} + y_i \cdot \mathbf{s}.$$

In the case $x_i = y_i$ this implies $\mathbf{x} = \mathbf{y}$, whereas in the case $x_i \neq y_i$ this implies $\mathbf{y} = \mathbf{x} + \mathbf{s}$.

(2) Let g be an arbitrary Simon function with period \mathbf{s} . We have to write g in the form $g = P \circ f_{\mathbf{s}}$. By (1), we know that $f_{\mathbf{s}}(\mathbf{x}) = f_{\mathbf{s}}(\mathbf{y})$ iff $\mathbf{y} \in \{\mathbf{x}, \mathbf{x} + \mathbf{s}\}$. So $f_{\mathbf{s}}$ and g already have the same arguments that collide. It remains to map $f_{\mathbf{s}}(\mathbf{x})$ to the correct image $g(\mathbf{x})$ via P . To this end define the bijection

$$P : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n, \quad \mathbf{x} \mapsto g(\mathbf{x} + x_i \mathbf{s}).$$

For all $\mathbf{x} \in \mathbb{F}_2^n$ we obtain

$$\begin{aligned} P \circ f_{\mathbf{s}}(\mathbf{x}) &= P(\mathbf{x} + x_i \mathbf{s}) = g(\mathbf{x} + x_i \mathbf{s} + (\mathbf{x} + x_i \mathbf{s})_i \cdot \mathbf{s}) \\ &= g(\mathbf{x} + x_i \mathbf{s} + (\mathbf{x} + 1)_i \cdot \mathbf{s}) = g(\mathbf{x} + \mathbf{s}) = g(\mathbf{x}), \end{aligned}$$

which implies $g = P \circ f_{\mathbf{s}}$. □

Instantiation of Function Choice. Throughout the paper, we instantiate our function $f_{\mathbf{s}}$ with the period $\mathbf{s} = (s_{n-1}, \dots, s_0) = 0^{n-2}11$ and $x_i = x_0$. We may realize $f_{\mathbf{s}}$ with n **cnot**-gates for copying \mathbf{x} , and an additional 2 **cnot**-gates for the controlled addition of \mathbf{s} via control bit 0. See Figure 4 for an implementation of $f_{\mathbf{s}}$ with $n = 3$.

Our function choice has the advantage that it can be implemented with only $n + 2$ **cnot** gates (if we are able to avoid **swaps**). In addition, we need $2n$ Hadamards for realizing SIMON. Thus we obtain a small circuit norm $CN = 10(n + 2) + 2n$, which in turn implies a relatively small error on IBM-Q16. We perform further circuit norm minimization in Section 3.2.

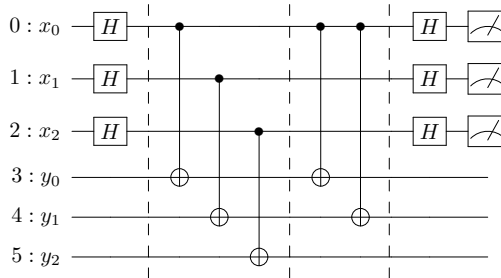


Fig. 4: Simon circuit Q_1 with our realization of f_s and $CN(Q_1) = 56$. The first 3 **cnots** copy \mathbf{x} , the remaining two **cnots** add $\mathbf{s} = 110$.

Discussion of our Simple Function Choice. As shown in Lemma 3.1, our function f_s is general in the sense that any Simon function is of the form $g = P \circ f_s(\mathbf{x})$. However, for obtaining small circuit norm we instantiate our Simon function with the simplest choice, where P is the identity function. In general, we could instantiate non-trivial P via some variable-length PRF with fixed key such as SiMeck [26]. This would however result in an explosion of the circuit norm and therefore in an explosion of IBM-Q16’s noise rate $\tau(n)$.

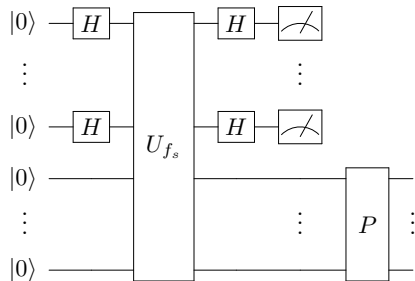


Fig. 5: SIMON with a general Simon function $P \circ f_s$.

controlled by P). So, quantumly the choice of a non-trivial P would just unnecessarily increase the error rate τ .

However, we would like to point out that choosing P as the identity function implies that classically extract the period \mathbf{s} is *not hard*. Notice that $f_s(\mathbf{x}) \in \{\mathbf{x}, \mathbf{x} + \mathbf{s}\}$. Thus, we may compute $f_s(1^n) + 1^n = \mathbf{s}$. The reason that $f_s(\mathbf{x})$ classically reveals its period so easily is that the image $\mathbf{x} + \mathbf{s}$ together with the argument \mathbf{x} directly gives us \mathbf{s} . This correlation between argument \mathbf{x} and image $\mathbf{x} + \mathbf{s}$ is destroyed by a random P , which explains why in general period finding classically becomes as hard as collision finding.

However, as explained above, SIMON does *not profit* from a trivial P , since SIMON is oblivious to concrete function values.

Thus, SIMON with a general Simon function could be implemented as depicted in Figure 5, where the permutation P is quantumly implemented in-place on the last n qubits (with at most one ancilla bit as shown in [22]). But already from Figure 5 one observes that P does not at all effect the SIMON algorithm. In fact, SIMON outputs the measurement of the first n qubits, which only depend on which arguments $\mathbf{x}, \mathbf{x} + \mathbf{s}$ collide under f_s , but *not* which function value $f_s(\mathbf{x}) = f_s(\mathbf{x} + \mathbf{s})$ they take (which is

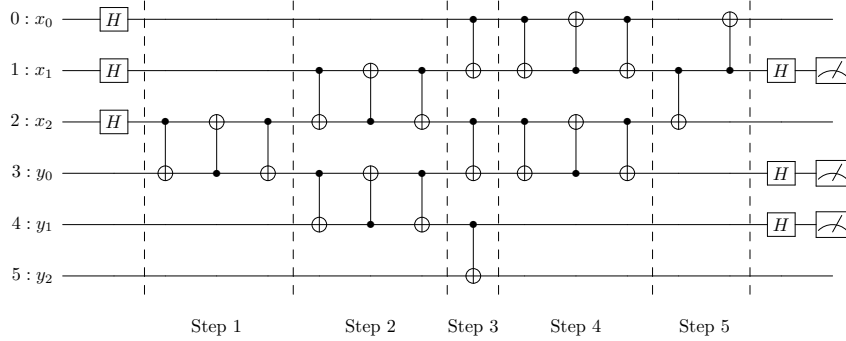


Fig. 6: IBM-Q16 compiles Q_1 to Q_2 with $CN(Q_2) = 206$.

From the discussion before, it should not be hard to see that Q_3 realizes Q_{fs}^{SIMON} , but yet it has to be optimized for IBM-Q16. First of all observe that **cnot** is self-inverse, and thus we can eliminate the two **cnot**(2, 3) gates. Afterwards, we can safely remove qubit 3. The resulting situation for qubits 0, 1, 2 is depicted in Figure 9, where we use a control bit change (see Figure 8).

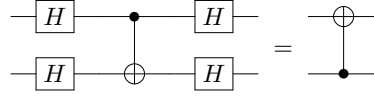


Fig. 8: Control bit change.

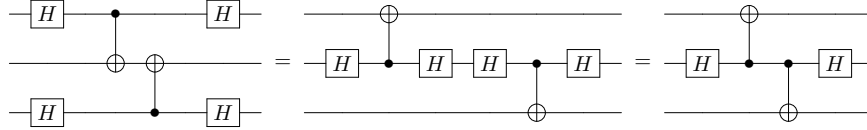


Fig. 9: Optimization of Q_3 .

From Figure 9 we see that the change of control bits from **cnot**(0, 1), **cnot**(2, 1) to **cnot**(1, 0), **cnot**(1, 2) leads to some cancellation of self-inverse Hadamard gates. Moreover, the second Hadamard of qubit 1 can be eliminated, since it does not influence the measurement. We end up with circuit Q_4 with an optimized gate count of $CN(Q_4) = 33$.

Since $\{0, 1\}, \{1, 2\}, \{4, 5\} \in E$, all three **cnots** of Q_4 can directly be realized on IBM-Q16. Notice that a configuration with optimal circuit norm is in general not unique. For our example, the following configuration yields the same circuit norm as the configuration of Q_4 :

$$3 : y_0 \quad 4 : x_0 \quad 5 : y_1 \quad 6 : x_1 \quad 8 : y_2 \quad 9 : x_2.$$

We optimized our IBM-Q16 implementation by choosing among all configurations with minimal circuit norm the one using IBM-Q16's qubits of smallest

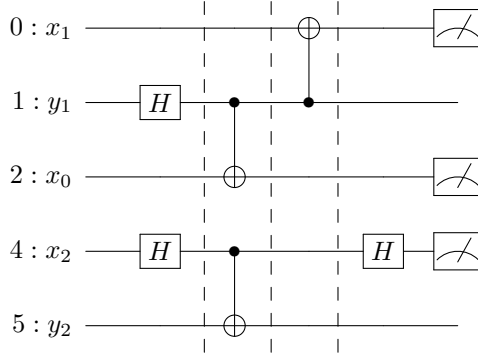


Fig. 10: Optimized circuit Q_4 on IBM-Q16 with $\text{CN}(Q_4) = 33$.

error rate (see Figure 2). The choice of our configurations is given in Table 1, a complete list of optimized circuits of this table can be found in Appendix A, Figure 17.

$n \backslash q$	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	CN
2	y_1	x_0					y_0								x_1	21
3	y_1	x_0					y_0		x_2	y_2					x_1	33
4	y_1	x_0			x_3	y_3	y_0		x_2	y_2					x_1	45
5	y_1	x_0			x_3	y_3	y_0		x_2	y_2	x_4	y_4			x_1	57
6	y_1	x_0			x_3	y_3	y_0		x_2	y_2	x_4	y_4	x_5	y_5	x_1	69
7	y_1	x_0	x_6	y_6	x_3	y_3	y_0		x_2	y_2	x_4	y_4	x_5	y_5	x_1	81

Table 1: Table of configurations.

3.3 Experiments on IBM Q 16

For each dimension $n = 2, \dots, 7$ we took 8192 measurements on IBM-Q16 of our optimized circuits from the previous section. The resulting relative frequencies are depicted in Figure 11. For each n , let $S(n)$ denote the set of erroneous measurements in $\mathbb{F}_2^n \setminus \mathbf{s}^\perp$. Then we compute the error rate $\tau(n)$ as $\tau(n) = \frac{|S(n)|}{8192}$. In Figure 11 we draw horizontal lines $\frac{1-\tau(n)}{2^{n-1}}$, respectively $\frac{\tau(n)}{2^{n-1}}$, for the prob-

ability distributions of our LSN Error Model for orthogonal, respectively non-orthogonal, vectors.

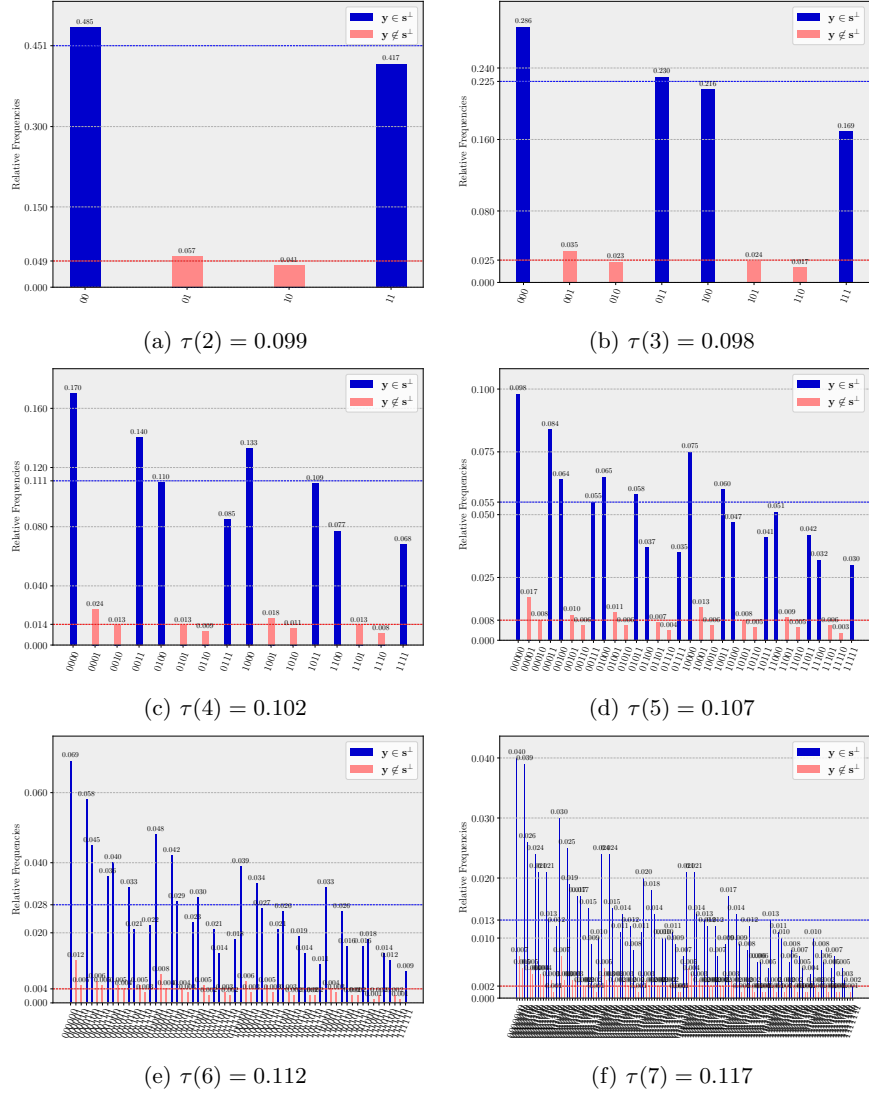


Fig. 11: IBM-Q16 measurements of our optimized circuits (see Appendix A, Figure 17).

On the positive side, we observe that vectors in \mathbf{s}^\perp are much more frequent. Hence, IBM-Q16 is noisy, but in principle works well for period finding. E.g. for $n = 3$, we have $\{\mathbf{s}\}^\perp = \{011\}^\perp = \{000, 011, 100, 111\}$, and we measure one of these vectors with probability $1 - \tau \approx 90\%$.

On the negative side, we observe the following effects.

- **Different qubit quality.** We deliberately ordered our qubits by error rate to make the quality effect visible. Using the IBM-Q16 calibration, we choose lowest error rate for the least significant bit x_0 up to highest error rate for the most significant bit x_{n-1} (nevertheless e.g. for $n = 4$ it seems that the qubit for x_2 performed worse than the one for x_3).
- **Bias towards 0.** In Figure 11 we ordered our measurements on the x -axis lexicographically. It can be observed that in general measurements with small Hamming weight appear with larger frequencies than large Hamming weight measurements. This indicates a bias towards the $|0\rangle$ qubit, which seems to be a natural physical effect since $|0\rangle$ is a non-activated ground state.
- **Increasing $\tau(n)$.** The error rate $\tau(n)$ is a function increasing in n . This is what we expected, since the circuit norm increases with n , and for larger n we also had to include lower quality qubits.

Remark 3.1. We experimented with different periodic f_s , especially more complex than our choice from Definition 3.2. Qualitatively, we observed similar effects albeit with larger error rates $\tau(n)$.

The effects of *different qubit quality* and *bias towards 0* obviously violate our LSN Error Model from Definition 2.2, since they destroy the uniform distribution among orthogonal, respectively non-orthogonal, vectors. However, we introduce in the subsequent Section 4 simple smoothing technique that (almost perfectly) mitigate both effects.

4 Smoothing Techniques

Let us first introduce a simple permutation technique that mitigates the *different qubit quality*.

Permutation Technique. We already saw in Section 3.2 that configurations for some quantum circuit C with minimal circuit norm are not unique. Let M be the set of configurations with minimal circuit norm, including all permutations of qubits. Then we may perform measurements for circuits randomly chosen from M , see Algorithm 2. This approach averages over the qubit quality, while due to its invariant circuit norm preserving the error rate $\tau(n)$.

Algorithm 2: Permutation Technique.

- 1 Let $M := \{\text{Configurations of } C \text{ with minimal circuit norm}\}$.
 - 2 Evaluate C with configurations chosen randomly from M .
-

Instantiation of M in our experiments. First we chose a set of of highest quality qubits $\{i_1, \dots, i_{2n-1}\}$ together with a starting configuration with minimal circuit norm. Let this be

$$i_1 : x_0 \quad i_2 : x_1 \quad i_3 : y_1 \quad i_4 : x_2 \quad \dots \quad i_{2n-2} : x_{n-1} \quad i_{2n-1} : y_{n-1}.$$

We then chose $b \sim \mathcal{U}$ and a random permutations π on $\{2, \dots, n-1\}$. This gives us circuit-norm preserving configurations

$$i_1 : x_b \quad i_2 : x_{1-b} \quad i_3 : y_1 \quad i_4 : x_{\pi(2)} \quad \dots \quad i_{2n-2} : x_{\pi(n-1)} \quad i_{2n-1} : y_{\pi(n-1)}.$$

We took 50 circuit-norm preserving configurations, and for each we performed 8192 measurements on IBM-Q16.

The experimental results of our Permutation Technique are illustrated for $n = 5$ in Figure 13b. In comparison, we have in Figure 13a the unsmoothed distribution for 8192 measurements of a single optimal configuration (as in Figure 11). We already see a significant distribution smoothing, especially vectors with the same Hamming weight obtain similar probabilities. But of course, there is still a clear *bias towards 0*, which cannot be mitigated by permutations.

Double-Flip Technique. To mitigate the effect that vectors with small Hamming weight are measured more frequently than vectors with large Hamming weight, we flip in Simon’s circuit all bits via NOT-gates X before measurement, see Figure 12. This flipping inverts the bias towards 0 that comes from the quantum measurement (not from the previous quantum computation). Since after flipping we measure the complement, we have to again flip all bits (classically) after measurement and combine them with the original measurements.

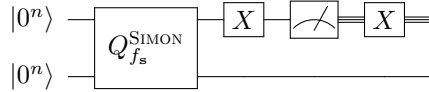
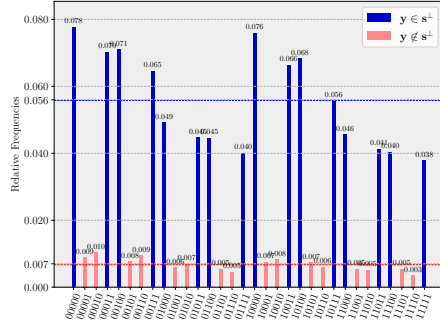


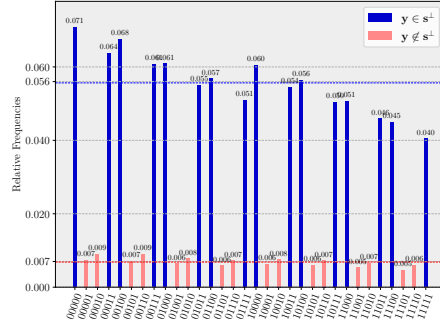
Fig. 12: Double-Flip circuit Q_{DF} . Triple lines represent classical wires.

Experimental Results and Discussion. We performed 8192 measurements with circuit Q_{DF} from Figure 12, the results are illustrated in Figure 13c. As expected, we now obtain a bias towards 1. Hence, in the *Double-Flip Technique* we put together the original measurements with 0-bias from Figure 13a and the flipped measurements with 1-bias from Figure 13c, resulting in the smoothed distribution from Figure 13d.

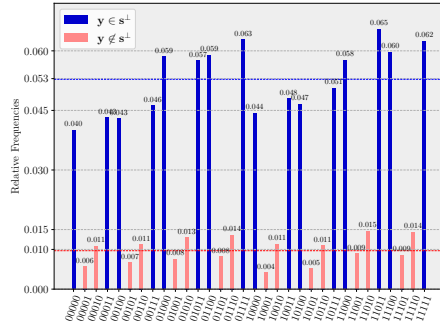
From Figure 13d we already see that the Double-Flip Technique is quite effective. Moreover, similar to the Permutation Technique, Double-Flip is a general smoothing technique that can be applied for other quantum circuits as well. However, there is also a significant drawback of Double-Flip, since it requires additional (small) quantum circuitry for performing X . Thus, as opposed to the Permutation Technique the Double-Flip does not preserve circuit norm. This implies that it slightly increases the error rate τ , as we will see in Section 4.1, where we study more closely the quality of our smoothing techniques.



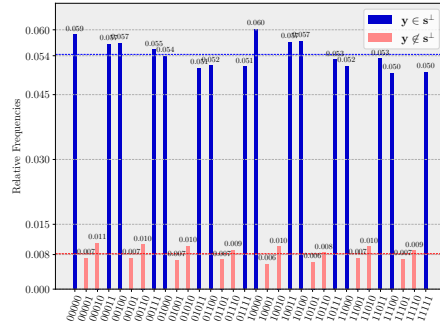
(a) Unsmoothed measurements, $n=5$.



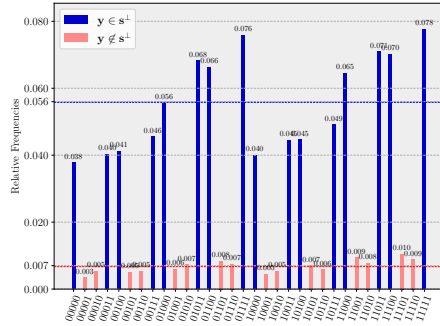
(b) Permutation Technique.



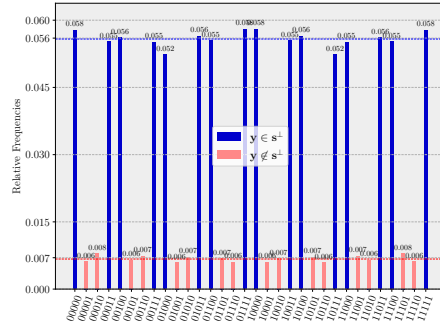
(c) Measurements Q_{DF} (Figure 12).



(d) Double-Flip Technique.



(e) Complemented measurements.



(f) Hamming Technique.

Fig. 13: Smoothed IBM-Q16 measurements.

Hamming Technique. The Hamming Technique is similar to the Double-Flip Technique, but as opposed to Double-Flip Hamming is specific to Simon-type problems and a purely classical post-processing of data without adding any additional circuitry.

Let $Q \subseteq \mathbb{F}_2^n$ be a multiset of quantum measurements, e.g. the set of 8192 measurements from Figure 13a. Then consider the complementary multiset

$$\bar{Q} = \{\mathbf{q} + \mathbf{1}^n \mid \mathbf{q} \in Q\},$$

where we flip all bits. Let $\mathbf{q} \in Q \cap \mathbf{s}^\perp$, i.e. \mathbf{q} is a measurement in the subspace orthogonal to \mathbf{s} . By complementing Q we want to preserve orthogonality, i.e. we want to have $\mathbf{q} + \mathbf{1}^n \in \mathbf{s}^\perp$ which is true iff $\mathbf{1}^n \in \mathbf{s}^\perp$ by the subspace structure.

Thus, complementation preserves orthogonality iff $\mathbf{1}^n \in \mathbf{s}^\perp$, which is in turn equivalent to even Hamming weight $h(\mathbf{s})$. Similar to Double-Flip, in the *Hamming Technique* we combine both measurements $Q \cup \bar{Q}$. The Hamming Technique mitigates the effect that for each $\mathbf{q} \in Q$ with large frequency (due to the 0-bias) we also obtain $\mathbf{q} + \mathbf{1}^n$ with small frequency (due to the 0-bias), and vice versa. Thus, averaging both frequencies should smooth our distribution closer to uniformity.

What happens if $\mathbf{1}^n \notin \mathbf{s}^\perp$? We want to add some vector $\mathbf{v} \in \mathbb{F}_2^n$ with Hamming weight as large as possible. It is not hard to see that there always exists some $\mathbf{v} \in \mathbf{s}^\perp$ with $h(\mathbf{v}) \geq n - 1$. Thus, we can simply try all $n + 1$ possible vectors.

Experimental Results. Since our instantiation of $f_{\mathbf{s}}$ from Section 3.1 uses even-weight periods \mathbf{s} , we can use the multiset \bar{Q} (with $\mathbf{1}^n$), which was done in Figure 13e and is a direct mirroring of Q in Figure 13a. The multiset of measurement $Q \cup \bar{Q}$ is then depicted in Figure 13f.

In comparison with Double-Flip from Figure 13d, we see that the Hamming technique provides in Figure 13f a distribution which is closer to the uniform distribution among orthogonal and non-orthogonal vectors. Thus, for our experimental data one should prefer the Hamming technique over Double-Flip.

Combination of techniques. The same preference can be observed when we combine the Permutation technique with either Double-Flip (see Figure 14a) or with Hamming (see Figure 14b).

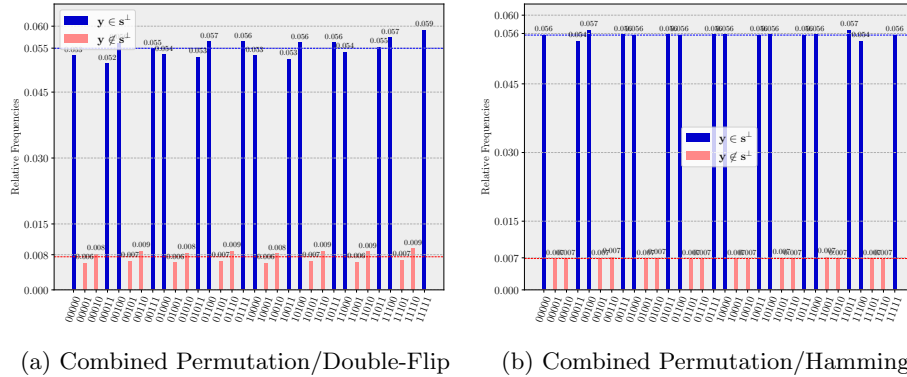


Fig. 14: Smoothing using a combination of techniques.

The combination Permutation/Hamming seems to outperform Permutation/Double-Flip, and Permutation/Hamming almost optimally follows our LSN Error Model from Definition 2.2.

4.1 Quality Measures Statistics.

Let us introduce a well-known statistical distance that quantitatively measures the effectiveness of our smoothing techniques. Recall that we require error distributions close to our LSN Error Model, in order to justify the proper use of LPN solvers in subsequent sections.

The Kullback-Leibler divergence describes the loss of information when going from a distribution P – e.g. our LSN Error Model distribution – to another distribution Q – e.g. our smoothed IBM-Q16 measurements.

Definition 4.1 (Kullback–Leibler divergence (KL)). *The Kullback-Leibler divergence of two probability distributions P towards Q on \mathbb{F}_2^n is*

$$D_{KL}(P||Q) := \sum_{y \in \mathbb{F}_2^n} P(Y) \log \left(\frac{P(y)}{Q(y)} \right) .$$

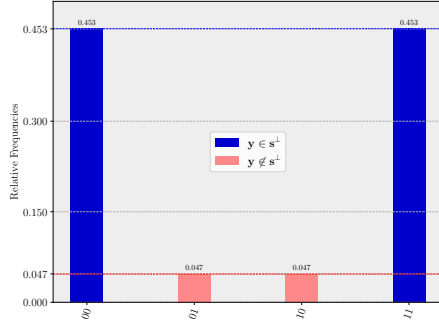
We compute KL and the error rate τ on the data from Figures 13 and 14. The results are given in Table 2.

Measure \ Smoothing	KL	τ
None	0.04644	0.10730
Permutation	0.01596	0.10954
Double-Flip	0.00600	0.13104
Permutation/Double-Flip	0.00297	0.12044
Hamming	0.00139	0.10730
Permutation/Hamming	0.00011	0.10954

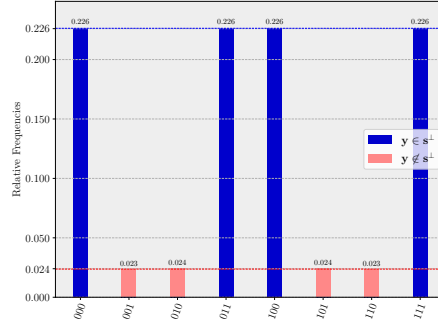
Table 2: Kullback–Leibler applied to our Smoothing Techniques.

As we would expect for KL, Hamming is more effective than Double-Flip. Also as predicted, Double-Flip increases the error rate τ , whereas the other techniques leave τ (basically) unchanged. In particular, Hamming leaves τ unchanged, since it is only a classical post-processing of our quantum data. We have already seen qualitatively in Figure 14 that the combination Permutation/Hamming performs best. This is supported also quantitatively in Table 2: KL is very close to zero, indicating that via Permutation/Hamming smoothed IBM-Q16 quantum measurements almost perfectly agree with the LSN Error Model.

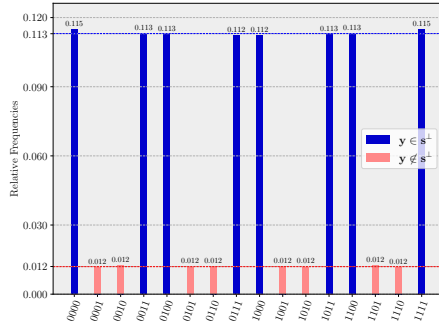
The results of applying Permutation/Hamming to all $n = 2, \dots, 7$ are depicted in Figure 15.



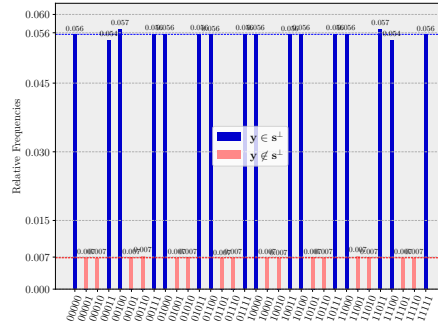
(a) $n = 2$, $\tau = 0.09347$, $KL = 0.00000$.



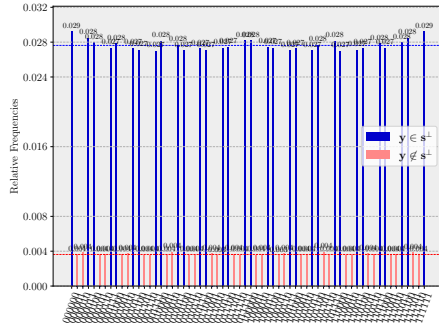
(b) $n = 3$, $\tau = 0.09479$, $KL = 0.00002$.



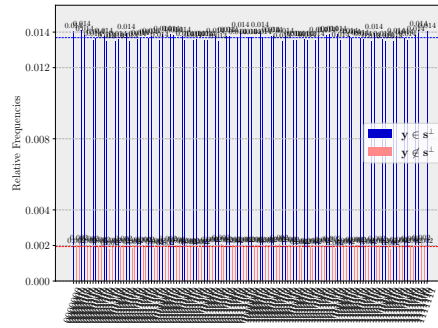
(c) $n = 4$, $\tau = 0.09546$, $KL = 0.00009$.



(d) $n = 5$, $\tau = 0.10954$, $KL = 0.00011$.



(e) $n = 6$, $\tau = 0.11602$, $KL = 0.00038$.



(f) $n = 7$, $\tau = 0.12398$, $KL = 0.00022$.

Fig. 15: Via Permutation/Hamming Technique smoothed IBM-Q16 measurements for $n = 2, \dots, 7$. KL is the Kullback-Leibler divergence to the LSN Error Model distribution.

5 LSN is Polynomial Time Equivalent to LPN

In the previous section, we smoothed our IBM-Q16 experiments to the LSN Error Model (Definition 2.2). Recall that the LSN Error Model states that with probability τ we measure in the quantum circuit $Q_{f_s}^{\text{SIMON}}$ some uniformly distributed $\mathbf{y} \in \mathbb{F}_2^n \setminus \mathbf{s}^\perp$. The question is now whether such erroneous \mathbf{y} as in our error model can easily be handled, i.e. whether LSN can be efficiently solved.

In this section, we answer this question in the negative. Namely, we show that solving LSN is tightly as hard as solving the well-studied LPN problem, which is supposed to be hard even on quantum computers.

Definition 5.1 (LPN-Problem). *Let $\mathbf{s} \in \mathbb{F}_2^n \setminus \{\mathbf{0}\}$ be chosen uniformly at random, and let $\tau \in [0, \frac{1}{2})$. In the Learning Parity with Noise problem, denoted $\text{LPN}_{n,\tau}$, one obtains access to an oracle $\mathcal{O}_{\text{LPN}}(\mathbf{s})$ that provides samples $(\mathbf{a}, \langle \mathbf{a}, \mathbf{s} \rangle + \epsilon)$, where $\mathbf{a} \sim \mathcal{U}_n$ and $\epsilon \sim \text{Ber}_\tau$. The goal is to compute \mathbf{s} .*

Definition 5.1 explicitly excludes $\mathbf{s} = \mathbf{0}$ in LPN. Notice that the case $\mathbf{s} = \mathbf{0}$ implies that the LPN oracle has distribution $U_n \times \text{Ber}_\tau$, whereas in the case $\mathbf{s} \neq \mathbf{0}$ we have $\mathbb{P}_{\mathbf{a}}[\langle \mathbf{a}, \mathbf{s} \rangle = 0] = \frac{1}{2}$ and therefore $\mathbb{P}_{\mathbf{a}}[\langle \mathbf{a}, \mathbf{s} \rangle + \epsilon = 0] = \frac{1}{2}$. Hence, for $\mathbf{s} \neq \mathbf{0}$ the LPN samples have distribution $U_n \times U$. This allows us to easily distinguish both cases by a majority test, whenever τ is polynomially bounded away from $\frac{1}{2}$. In conclusion, $\mathbf{s} = \mathbf{0}$ is not a hard case for LPN and may wlog be excluded.

Let us now define the related *Learning Simon with Noise* problem that reflects the LSN Error Model.

Definition 5.2 (LSN-Problem). *Let $\mathbf{s} \in \mathbb{F}_2^n \setminus \{\mathbf{0}\}$ be chosen uniformly at random, and let $\tau \in [0, \frac{1}{2})$. In the Learning Simon with Noise problem, denoted $\text{LSN}_{n,\tau}$, one obtains access to an oracle $\mathcal{O}_{\text{LSN}}(\mathbf{s})$ that provides samples \mathbf{y} , where $\mathbf{y} \in \mathbb{F}_2^n$ is distributed as in Definition 2.2, i.e.*

$$\mathbb{P}[\mathbf{y}] = \begin{cases} \frac{1-\tau}{2^{n-1}} & , \text{ if } \mathbf{y} \in \mathbf{s}^\perp \\ \frac{\tau}{2^{n-1}} & , \text{ else} \end{cases} \quad \text{and therefore } \mathbb{P}[\langle \mathbf{y}, \mathbf{s} \rangle = 0] = 1 - \tau.$$

The goal is to compute \mathbf{s} .

In the following we prove that $\text{LSN}_{n,\tau}$ is polynomial time equivalent to $\text{LPN}_{n,\tau}$ by showing that we can perfectly mutually simulate $\mathcal{O}_{\text{LPN}}(\mathbf{s})$ and $\mathcal{O}_{\text{LSN}}(\mathbf{s})$. The purpose of excluding $\mathbf{s} \neq \mathbf{0}$ from $\text{LPN}_{n,\tau}$ is to guarantee in the reduction non-trivial periods $\mathbf{s} \neq \mathbf{0}$ in $\text{LSN}_{n,\tau}$.

Theorem 5.1 (Equivalence of LPN and LSN). *Let \mathcal{A} be an algorithm that solves $\text{LPN}_{n,\tau}$ (respectively $\text{LSN}_{n,\tau}$) using m oracle queries in time T with success probability $\epsilon_{\mathcal{A}}$. Then there exists an algorithm \mathcal{B} that solves $\text{LSN}_{n,\tau}$ (respectively $\text{LPN}_{n,\tau}$) using m oracle queries in time T with success probability $\epsilon_{\mathcal{B}} \geq \frac{\epsilon_{\mathcal{A}}}{2}$.*

Algorithm 3: LPN \Rightarrow LSN

Input : $n, \tau, \mathcal{O}_{\text{LSN}}(\mathbf{s}), m$
Output: \mathbf{s}
1 Choose $\mathbf{z} \sim \mathcal{U}_n$.
2 **for** $i = 1$ **to** m **do**
3 Set $\mathbf{y}_i \leftarrow \mathcal{O}_{\text{LSN}}(\mathbf{s})$.
4 Choose $b_i \sim \mathcal{U}$.
5 **end**
6 $\mathbf{s} \leftarrow \mathcal{A}_{\text{LPN}}(n, \tau, (\mathbf{y}_1 + b_1 \mathbf{z}, b_1), \dots, (\mathbf{y}_m + b_m \mathbf{z}, b_m))$

Proof. Assume that we want to solve LSN via an algorithm \mathcal{A}_{LPN} with success probability $\epsilon_{\mathcal{A}}$ as in Algorithm 3.

We show in the following that Algorithm 3 perfectly simulates the oracle $\mathcal{O}_{\text{LPN}}(\mathbf{s})$ via $\mathcal{O}_{\text{LSN}}(\mathbf{s})$ if the vector $\mathbf{z} \sim \mathcal{U}_n$ chosen in Line 1 satisfies $\langle \mathbf{z}, \mathbf{s} \rangle = 1$. Since $\mathbf{s} \neq \mathbf{0}$, we have $\mathbb{P}_{\mathbf{z}}[\langle \mathbf{z}, \mathbf{s} \rangle = 1] = \frac{1}{2}$. Therefore Algorithm 3 succeeds with probability

$$\epsilon_{\mathcal{B}} \geq \mathbb{P}_{\mathbf{z}}[\langle \mathbf{z}, \mathbf{s} \rangle = 1 \cap \mathcal{A} \text{ outputs } \mathbf{s}] = \frac{\epsilon_{\mathcal{A}}}{2}.$$

Let us now show correctness of Algorithm 3. We first show that the constructed LPN samples $(\mathbf{y} + b\mathbf{z}, b)$ have the correct distribution. Let $\epsilon = \langle \mathbf{y} + b\mathbf{z}, \mathbf{s} \rangle + b$. Since $\langle \mathbf{z}, \mathbf{s} \rangle = 1$, we have

$$\mathbb{P}_{\mathbf{y}}[\epsilon = 1] = \mathbb{P}_{\mathbf{y}}[\langle \mathbf{y} + b\mathbf{z}, \mathbf{s} \rangle + b = 1] = \mathbb{P}_{\mathbf{y}}[\langle \mathbf{y}, \mathbf{s} \rangle + b\langle \mathbf{z}, \mathbf{s} \rangle + b = 1] = \mathbb{P}_{\mathbf{y}}[\langle \mathbf{y}, \mathbf{s} \rangle = 1] = \tau.$$

It remains to show that $\mathbf{y} + b\mathbf{z}$ is uniformly distributed. To this end, we show that

$$p_0 = \mathbb{P}_{\mathbf{y}, b}[\mathbf{y} + b\mathbf{z} \mid \langle \mathbf{y}, \mathbf{s} \rangle = 0] = \frac{1}{2^n}.$$

Analogous, it follows that $p_1 = \mathbb{P}_{\mathbf{y}, b}[\mathbf{y} + b\mathbf{z} \mid \langle \mathbf{y}, \mathbf{s} \rangle = 1] = \frac{1}{2^n}$. From both statements we obtain

$$\mathbb{P}_{\mathbf{y}, b}[\mathbf{y} + b\mathbf{z}] = \mathbb{P}_{\mathbf{y}}[\langle \mathbf{y}, \mathbf{s} \rangle = 0] \cdot p_0 + \mathbb{P}_{\mathbf{y}}[\langle \mathbf{y}, \mathbf{s} \rangle = 1] \cdot p_1 = \frac{1 - \tau}{2^n} + \frac{\tau}{2^n} = \frac{1}{2^n},$$

as desired. It remains to show that

$$\begin{aligned} p_0 &= \mathbb{P}_{\mathbf{y}, b}[\mathbf{y} + b\mathbf{z} \mid \langle \mathbf{y}, \mathbf{s} \rangle = 0] \\ &= \mathbb{P}_b[b = 0] \cdot \mathbb{P}_{\mathbf{y}}[\mathbf{y} \mid \langle \mathbf{y}, \mathbf{s} \rangle = 0] + \mathbb{P}_b[b = 1] \cdot \mathbb{P}_{\mathbf{a}}[\mathbf{y} + \mathbf{z} \mid \langle \mathbf{y}, \mathbf{s} \rangle = 0] \\ &= \frac{1}{2} \left(\frac{1 - \tau}{2^{n-1}} + \frac{\tau}{2^{n-1}} \right) = \frac{1}{2^n}. \end{aligned}$$

This completes the analysis of Algorithm 3.

For Algorithm 4 we conclude the success probability analogous to the reasoning for Algorithm 3, i.e. we succeed when $\langle \mathbf{z}, \mathbf{s} \rangle = 1$ and \mathcal{A}_{LSN} succeeds. So let us assume in the following correctness analysis that we are in the case $\langle \mathbf{z}, \mathbf{s} \rangle = 1$. This implies for the constructed LSN samples $\mathbf{a} + b\mathbf{z}$ that

$$\langle \mathbf{a} + b\mathbf{z}, \mathbf{s} \rangle = 0 \Leftrightarrow \langle \mathbf{a}, \mathbf{s} \rangle + b\langle \mathbf{z}, \mathbf{s} \rangle = 0 \Leftrightarrow \langle \mathbf{a}, \mathbf{s} \rangle = b.$$

Algorithm 4: LSN \Rightarrow LPN

Input : $n, \tau, \mathcal{O}_{\text{LPN}}(\mathbf{s}), m$
Output: \mathbf{s}
1 Choose $\mathbf{z} \sim \mathcal{U}_n$.
2 **for** $i = 1$ **to** m **do**
3 | Set $(\mathbf{a}_i, b_i) \leftarrow \mathcal{O}_{\text{LPN}}(\mathbf{s})$.
4 **end**
5 $\mathbf{s} \leftarrow \mathcal{A}_{\text{LSN}}(n, \tau, \mathbf{a}_1 + b_1\mathbf{z}, \dots, \mathbf{a}_m + b_m\mathbf{z})$

Let $\epsilon = \langle \mathbf{a}, \mathbf{s} \rangle + b$. It follows that

$$\mathbb{P}_{\mathbf{a},b}[\langle \mathbf{a} + b\mathbf{z}, \mathbf{s} \rangle = 0] = \mathbb{P}_{\mathbf{a},b}[\langle \mathbf{a}, \mathbf{s} \rangle = b] = \mathbb{P}_{\mathbf{a},b}[\epsilon = 0] = 1 - \tau.$$

We also have to show that we obtain a uniform distribution among all $\mathbf{a} + b\mathbf{z} \in \mathbf{s}^\perp$. This follows from

$$\begin{aligned} \mathbb{P}_{\mathbf{a},b}[\mathbf{a} + b\mathbf{z} \mid \langle \mathbf{a} + b\mathbf{z}, \mathbf{s} \rangle = 0] &= \mathbb{P}_{\mathbf{a},b}[\mathbf{a} + b\mathbf{z} \mid \langle \mathbf{a}, \mathbf{s} \rangle = b] \\ &= \mathbb{P}_{\mathbf{a}}[\langle \mathbf{a}, \mathbf{s} \rangle = 0] \cdot \mathbb{P}_{\mathbf{a},b}[\mathbf{a} + b\mathbf{z} \mid \langle \mathbf{a}, \mathbf{s} \rangle = b = 0] + \\ &\quad \mathbb{P}_{\mathbf{a}}[\langle \mathbf{a}, \mathbf{s} \rangle = 1] \cdot \mathbb{P}_{\mathbf{a},b}[\mathbf{a} + b\mathbf{z} \mid \langle \mathbf{a}, \mathbf{s} \rangle = b = 1] \\ &= \frac{1}{2} \cdot \mathbb{P}_{\mathbf{a}}[\mathbf{a} \mid \langle \mathbf{a}, \mathbf{s} \rangle = 0] + \frac{1}{2} \cdot \mathbb{P}_{\mathbf{a}}[\mathbf{a} + \mathbf{z} \mid \langle \mathbf{a}, \mathbf{s} \rangle = 1] \\ &= \frac{1}{2} \cdot \frac{1}{2^{n-1}} + \frac{1}{2} \cdot \frac{1}{2^{n-1}} = \frac{1}{2^{n-1}}. \end{aligned}$$

Analogous, we can show that we obtain a uniform distribution among all $\mathbf{a} + b\mathbf{z} \in \mathbb{F}_2^n \setminus \mathbf{s}^\perp$. This proves that we perfectly simulate LSN-samples via \mathcal{O}_{LPN} , and thus shows correctness of Algorithm 4. \square

Theorem 5.1 shows that under the LPN assumption we cannot expect to solve LSN in polynomial time. However, it does not exclude that quantum measurements that lead to an LSN distribution are still useful in the sense that they help us to solve period finding faster than on classical computers. In the following section, we show that LSN distributed quantum outputs indeed lead to speedups even for large error rates τ .

6 Theoretical Error Handling for Simon's Algorithm

It is well-known [19] that period finding for n -bit Simon functions classically requires time $\Omega(2^{\frac{n}{2}})$. So despite the hardness results of Section 5 we may still hope that even error-prone quantum measurements lead to period finding speedups. Indeed, it is also known that for any fixed $\tau < \frac{1}{2}$ the BKW algorithm [4] solves $\text{LPN}_{n,\tau}$ — and thus by Theorem 5.1 also $\text{LSN}_{n,\tau}$ — in time $2^{\mathcal{O}(\frac{n}{\log n})}$. This implies that asymptotically the combination of LSN samples together with a suitable LPN-solver already outperforms classical period finding.

In this work, we focus on the LPN-solvers of Esser, Kübler, May [11] rather than the class of BKW-type solvers [4,13,16,10], since they have a simple description and runtime analysis, are easy to implement, have low memory consumption, are sufficiently powerful for showing quantum advantage even for large errors $\tau < \frac{1}{2}$, and finally they are practically best for the IBM-Q16 error rates $\tau \in [0.09, 0.13]$.

We start with the analysis of the POOLED GAUSS algorithm [11]. POOLED GAUSS solves $\text{LPN}_{n,\tau}$ in time $\tilde{\Theta}\left(2^{\log(\frac{1}{1-\tau}) \cdot n}\right)$ using $\tilde{\Theta}(n^2)$ samples.

The following theorem shows that period finding with error-prone quantum samples in combination with POOLED GAUSS is superior to purely classical period finding whenever the error τ is bounded by $\tau \leq 0.293$.

Theorem 6.1. *In the LSN Error Model (Definition 2.2), POOLED GAUSS finds the period $\mathbf{s} \in \mathbb{F}_2^n$ of a Simon function $f_{\mathbf{s}}$ using $\tilde{\Theta}(n^2)$ many $\text{LSN}_{n,\tau}$ -samples, coming from practical measurements of Simon's circuit $Q_{f_{\mathbf{s}}}^{\text{SIMON}}$ with error rate τ , in time $\tilde{\Theta}\left(2^{\log(\frac{1}{1-\tau}) \cdot n}\right)$. This improves over classical period finding for error rates*

$$\tau < 1 - \frac{1}{\sqrt{2}} \approx 0.293.$$

Proof. We use Algorithm 3, where any $\mathcal{O}_{\text{LPN}}(\mathbf{s})$ -call is provided by a measurement of $Q_{f_{\mathbf{s}}}^{\text{SIMON}}$. In the LSN Error Model, this gives us an $\text{LSN}_{n,\tau}$ -instance which is transformed by Algorithm 3 into an $\text{LPN}_{n,\tau}$ -instance. We use POOLED GAUSS as the LPN-solver \mathcal{A}_{LPN} inside Algorithm 3. This immediately implies time complexity $\tilde{\Theta}\left(2^{\log(\frac{1}{1-\tau}) \cdot n}\right)$.

It remains to show outperformance of the classical algorithm, i.e. $\log\left(\frac{1}{1-\tau}\right) < \frac{1}{2}$. Notice that our condition $\tau < 1 - \frac{1}{\sqrt{2}}$ implies that $\frac{1}{1-\tau} < \sqrt{2}$ and therefore

$$\log\left(\frac{1}{1-\tau}\right) < \log(\sqrt{2}) = \frac{1}{2}.$$

□

Theorem 6.1 already shows the usefulness of a quite limited quantum oracle that only allows us polynomially many measurements, whenever its error rate τ is small enough.

If we allow for more quantum measurements, the WELL-POOLED GAUSS algorithm [11] solves $\text{LPN}_{n,\tau}$ in improved time and query complexity $\tilde{\Theta}(2^{f(\tau)n})$, where $f(\tau) = 1 - \frac{1}{1+\log(\frac{1}{1-\tau})}$. The following theorem shows that WELL-POOLED GAUSS in combination with error-prone quantum measurements improves on classical period finding for *any* error rate τ .

Theorem 6.2. *In the LSN Error Model (Definition 2.2), WELL POOLED GAUSS finds the period $\mathbf{s} \in \mathbb{F}_2^n$ of a Simon function $f_{\mathbf{s}}$ using $\tilde{\Theta}(2^{f(\tau)n})$ many $\text{LSN}_{n,\tau}$ -samples, coming from practical measurements of Simon's circuit $Q_{f_{\mathbf{s}}}^{\text{SIMON}}$ with*

error rate τ , in time $\tilde{\Theta}(2^{f(\tau)^n})$, where

$$f(\tau) = 1 - \frac{1}{1 + \log(\frac{1}{1-\tau})}.$$

This improves over classical period finding for all error rates $\tau < \frac{1}{2}$.

Proof. As in the proof of Theorem 6.1 we use Algorithm 3, where measurements of $Q_{f_s}^{\text{SIMON}}$ provide the $\mathcal{O}_{\text{LPN}}(\mathbf{s})$ -calls and WELL POOLED GAUSS is the LPN-solver \mathcal{A}_{LPN} . Correctness and the claimed complexities follow immediately.

It remains to show outperformance of any classical period finding algorithm. Notice that $\tau < \frac{1}{2}$ implies $\frac{1}{1-\tau} < 2$ and therefore $\log(\frac{1}{1-\tau}) < 1$. This in turn implies

$$f(\tau) = 1 - \frac{1}{1 + \log(\frac{1}{1-\tau})} < 1 - \frac{1}{2} = \frac{1}{2}.$$

□

The results of Theorem 6.1 and Theorem 6.2 show that quantum measurements of $Q_{f_s}^{\text{SIMON}}$ help us (asymptotically) even for large error rates τ , provided that our error model is sufficiently accurate.

7 Practical Error Handling for Simon's Algorithm

In this section, we compare the practical runtimes needed to find periods \mathbf{s} with the smoothed experimental data from our IBM-Q16 quantum measurements (see Figure 15) with purely classical period finding.

Notice that our LPN-solvers incur some polynomial overhead, which makes them for very small n as on IBM-Q16 inferior to purely classical period finding. Moreover, we would like to stress the experimental result of Section 3 that IBM-Q16's error rate $\tau(n)$ is a function increasing in n . So even if asymptotically LPN-solvers outperform classical period finding, a fast convergence of $\tau(n)$ towards $\frac{1}{2}$ prevents practical quantum advantage.

Periods classically. Let us start with the description of an optimal classical period finding algorithm, inspired by [19]. Naively, one may think that it is optimal to query f_s at different random points \mathbf{x}_i , until one hits the first collision $f_s(\mathbf{x}_i) = f_s(\mathbf{x}_j)$. However, assume that we have already queried the set of points $P = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\}$, without obtaining a collision. This gives us the information that \mathbf{s} is *not* in set of distances $D = \{\mathbf{x}_1 + \mathbf{x}_2, \mathbf{x}_2 + \mathbf{x}_3, \mathbf{x}_1 + \mathbf{x}_3\}$. This implies that we should not ask $\mathbf{x}_1 + \mathbf{x}_2 + \mathbf{x}_3$, since it lies at distance $\mathbf{x}_1 + \mathbf{x}_2$ of \mathbf{x}_3 . Hence an optimal algorithm keeps track of the set D of all excluded distances. This is realized in our algorithm PERIOD, see Algorithm 5.

Algorithm 5: PERIOD

Input : Access to f_s .
Output: Secret s .

```
1 begin
2   Set  $P = \{(\mathbf{0}, f_s(\mathbf{0}))\}$ . ▷ Set of queried points.
3   Set  $D = \{\mathbf{0}\}$ . ▷ Set of distances.
4   repeat
5     Select  $\mathbf{x} \in \operatorname{argmax}\{|\{s' \in D \mid (\mathbf{x} + s', \cdot) \notin P\}|\}$ . ▷ Optimal next query.
6      $P := P \cup (\mathbf{x}, f_s(\mathbf{x}))$  ▷ Update queries.
7     for  $(\mathbf{x}', f_s(\mathbf{x}')) \in P$  do
8       |  $D := D \cup \{\mathbf{x} + \mathbf{x}'\}$  ▷ Update distances.
9     end
10  until  $\exists \mathbf{x}' \neq \mathbf{x} : (\mathbf{x}', f_s(\mathbf{x})) \in P$  or  $|D| = 2^n - 1$ 
11  if  $|D| = 2^n - 1$  then return  $s \in \mathbb{F}_2^n \setminus D$ . ▷ Only possible period.
12  else return  $\mathbf{x} + \mathbf{x}'$ . ▷ Collision found.
13 end
```

Periods quantumly. By the result of Section 5 we may first transform our quantum measurements into LPN samples, and then use one of the LPN-solvers from Section 6. Since the error rates from our *smoothed* IBM-Q16 measurements (Figure 15) are below $\frac{1}{8}$, according to Theorem 6.1 we may use POOLED GAUSS.

Instead of applying the LSN-to-LPN reduction to our smoothed data, we directly adapt POOLED GAUSS into an LSN-solver, called POOLED LSN (Algorithm 6). POOLED LSN can be considered as a fault-tolerant version of SIMON (Algorithm 1) that iterates until we obtain an error-free set of $n - 1$ linearly independent vectors. Notice that error-freeness can be tested, since the resulting potential period \mathbf{s}' is correct iff $f_s(\mathbf{s}') = f_s(\mathbf{0})$.

Algorithm 6: POOLED LSN

Input : Pool $P \subset \mathbb{F}_2^n$ of LSN samples with $|P| \geq n - 1$
Output: Secret s

```
1 begin
2   repeat
3     Randomly select a linearly independent set  $Y = \{\mathbf{y}_1, \dots, \mathbf{y}_{n-1}\} \subseteq P$ .
4     Compute the unique  $\mathbf{s}' \in Y^\perp \setminus \{\mathbf{0}\}$ .
5     until  $f_s(\mathbf{s}') \stackrel{?}{=} f_s(\mathbf{0})$ 
6     return  $\mathbf{s}'$ .
7 end
```

Run time comparison. PERIOD and POOLED LSN exponentially often iterate their **repeat**-loops, where each iteration runs in polynomial time (using the right data structure). Hence, asymptotically the number of iterations dom-

inate runtimes for both algorithms. For ease of simplicity, we take as cost measure only the exponential number of loops, ignoring all polynomial factors (the polynomial factors actually dominate in practice for our small dimensions n). Using this (over-)simplified loop cost measure, we ran 10.000 iterations of PERIOD for $n = 2, \dots, 7$ and averaged over the runtimes. For the quantum period finding we took as pool P the complete smoothed data of Figure 15. We then also ran 10.000 iterations of POOLED LSN for $n = 2, \dots, 7$ and averaged over the runtimes. The resulting log-scaled runtimes are depicted in Figure 16.

As expected, PERIOD’s experimental runtime exponent is $\frac{n}{2}$. For POOLED LSN, we obtain an experimental regression line of roughly $\frac{n}{3}$, where the slope seems to decrease with n . This results in a cut-off point for the loop numbers between $n = 4$ and $n = 5$. Thus, experimentally we obtain quantum advantage, at least for our loop cost measure.

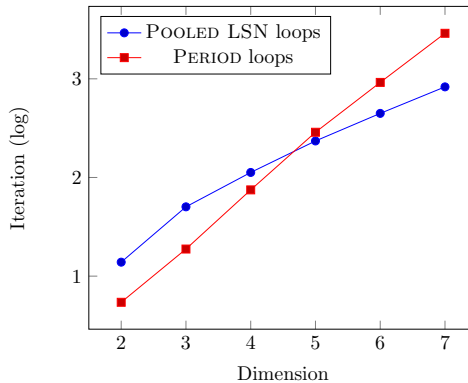


Fig. 16: Log-scaled loop iterations of POOLED LSN and PERIOD, averaged over 10.000 iterations.

Acknowledgement. We acknowledge use of the IBM Q for this work. The views expressed are those of the authors and do not reflect the official policy or position of IBM or the IBM Q team.

References

- 15-qubit backend: IBM Q team, "IBM Q 16 Melbourne backend specification V2.0.1," (2020). Retrieved from <https://quantum-computing.ibm.com>. Accessed 14. January 2020.
- Alagic, G., Russell, A.: Quantum-secure symmetric-key cryptography based on hidden shifts. In: Coron, J., Nielsen, J.B. (eds.) EUROCRYPT 2017, Part III. LNCS, vol. 10212, pp. 65–93. Springer, Heidelberg (Apr / May 2017)
- Alekhovich, M.: More on average case vs approximation complexity. In: 44th FOCS. pp. 298–307. IEEE Computer Society Press (Oct 2003)
- Blum, A., Kalai, A., Wasserman, H.: Noise-tolerant learning, the parity problem, and the statistical query model. In: 32nd ACM STOC. pp. 435–440. ACM Press (May 2000)
- Boneh, D., Zhandry, M.: Secure signatures and chosen ciphertext security in a quantum computing world. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part II. LNCS, vol. 8043, pp. 361–379. Springer, Heidelberg (Aug 2013)
- Bonnetain, X.: Quantum key-recovery on full AEZ. In: Adams, C., Camenisch, J. (eds.) SAC 2017. LNCS, vol. 10719, pp. 394–406. Springer, Heidelberg (Aug 2017)

7. Bonnetain, X., Hosoyamada, A., Naya-Plasencia, M., Sasaki, Y., Schrottenloher, A.: Quantum attacks without superposition queries: the offline simon’s algorithm. In: *Advances in Cryptology - ASIACRYPT 2019* (2019)
8. Bonnetain, X., Naya-Plasencia, M.: Hidden shift quantum cryptanalysis and implications. In: Peyrin, T., Galbraith, S. (eds.) *ASIACRYPT 2018, Part I*. LNCS, vol. 11272, pp. 560–592. Springer, Heidelberg (Dec 2018)
9. Calderbank, A.R., Rains, E.M., Shor, P.W., Sloane, N.J.: Quantum error correction and orthogonal geometry. *Physical Review Letters* 78(3), 405 (1997)
10. Esser, A., Heuer, F., Kübler, R., May, A., Sohler, C.: Dissection-BKW. In: Shacham, H., Boldyreva, A. (eds.) *CRYPTO 2018, Part II*. LNCS, vol. 10992, pp. 638–666. Springer, Heidelberg (Aug 2018)
11. Esser, A., Kübler, R., May, A.: LPN decoded. In: Katz, J., Shacham, H. (eds.) *CRYPTO 2017, Part II*. LNCS, vol. 10402, pp. 486–514. Springer, Heidelberg (Aug 2017)
12. Even, S., Mansour, Y.: A construction of a cipher from a single pseudorandom permutation. In: Imai, H., Rivest, R.L., Matsumoto, T. (eds.) *ASIACRYPT’91*. LNCS, vol. 739, pp. 210–224. Springer, Heidelberg (Nov 1993)
13. Guo, Q., Johansson, T., Löndahl, C.: Solving LPN using covering codes. In: Sarkar, P., Iwata, T. (eds.) *ASIACRYPT 2014, Part I*. LNCS, vol. 8873, pp. 1–20. Springer, Heidelberg (Dec 2014)
14. Hosoyamada, A., Sasaki, Y.: Cryptanalysis against symmetric-key schemes with online classical queries and offline quantum computations. In: Smart, N.P. (ed.) *CT-RSA 2018*. LNCS, vol. 10808, pp. 198–218. Springer, Heidelberg (Apr 2018)
15. Kaplan, M., Leurent, G., Leverrier, A., Naya-Plasencia, M.: Breaking symmetric cryptosystems using quantum period finding. In: Robshaw, M., Katz, J. (eds.) *CRYPTO 2016, Part II*. LNCS, vol. 9815, pp. 207–237. Springer, Heidelberg (Aug 2016)
16. Kirchner, P., Fouque, P.A.: An improved BKW algorithm for LWE with applications to cryptography and lattices. In: Gennaro, R., Robshaw, M.J.B. (eds.) *CRYPTO 2015, Part I*. LNCS, vol. 9215, pp. 43–62. Springer, Heidelberg (Aug 2015)
17. Kuwakado, H., Morii, M.: Security on the quantum-type even-mansour cipher. In: *Proceedings of the International Symposium on Information Theory and its Applications, ISITA 2012, Honolulu, HI, USA, October 28-31, 2012*. pp. 312–316 (2012), <http://ieeexplore.ieee.org/document/6400943/>
18. Leander, G., May, A.: Grover meets simon - quantumly attacking the FX-construction. In: Takagi, T., Peyrin, T. (eds.) *ASIACRYPT 2017, Part II*. LNCS, vol. 10625, pp. 161–178. Springer, Heidelberg (Dec 2017)
19. Montanaro, A., de Wolf, R.: A survey of quantum property testing. *Theory of Computing, Graduate Surveys* 7, 1–81 (2016), <https://doi.org/10.4086/toc.gs.2016.007>
20. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: Gabow, H.N., Fagin, R. (eds.) *37th ACM STOC*. pp. 84–93. ACM Press (May 2005)
21. Santoli, T., Schaffner, C.: Using simon’s algorithm to attack symmetric-key cryptographic primitives. *Quantum Information & Computation* 17(1&2), 65–78 (2017), <http://www.rintonpress.com/xxqic17/qic-17-12/0065-0078.pdf>
22. Shende, V.V., Prasad, A.K., Markov, I.L., Hayes, J.P.: Synthesis of reversible logic circuits. *IEEE Trans. on CAD of Integrated Circuits and Systems* 22(6), 710–722 (2003), <https://doi.org/10.1109/TCAD.2003.811448>

23. Shor, P.W.: Algorithms for quantum computation: Discrete logarithms and factoring. In: 35th FOCS. pp. 124–134. IEEE Computer Society Press (Nov 1994)
24. Simon, D.R.: On the power of quantum computation. In: 35th FOCS. pp. 116–123. IEEE Computer Society Press (Nov 1994)
25. Tame, M.S., Bell, B.A., Di Franco, C., Wadsworth, W.J., Rarity, J.G.: Experimental realization of a one-way quantum computer algorithm solving simon’s problem. *Phys. Rev. Lett.* 113, 200501 (Nov 2014), <https://link.aps.org/doi/10.1103/PhysRevLett.113.200501>
26. Yang, G., Zhu, B., Suder, V., Aagaard, M.D., Gong, G.: The simeck family of lightweight block ciphers. In: Güneysu, T., Handschuh, H. (eds.) CHES 2015. LNCS, vol. 9293, pp. 307–329. Springer, Heidelberg (Sep 2015)

A Appendix

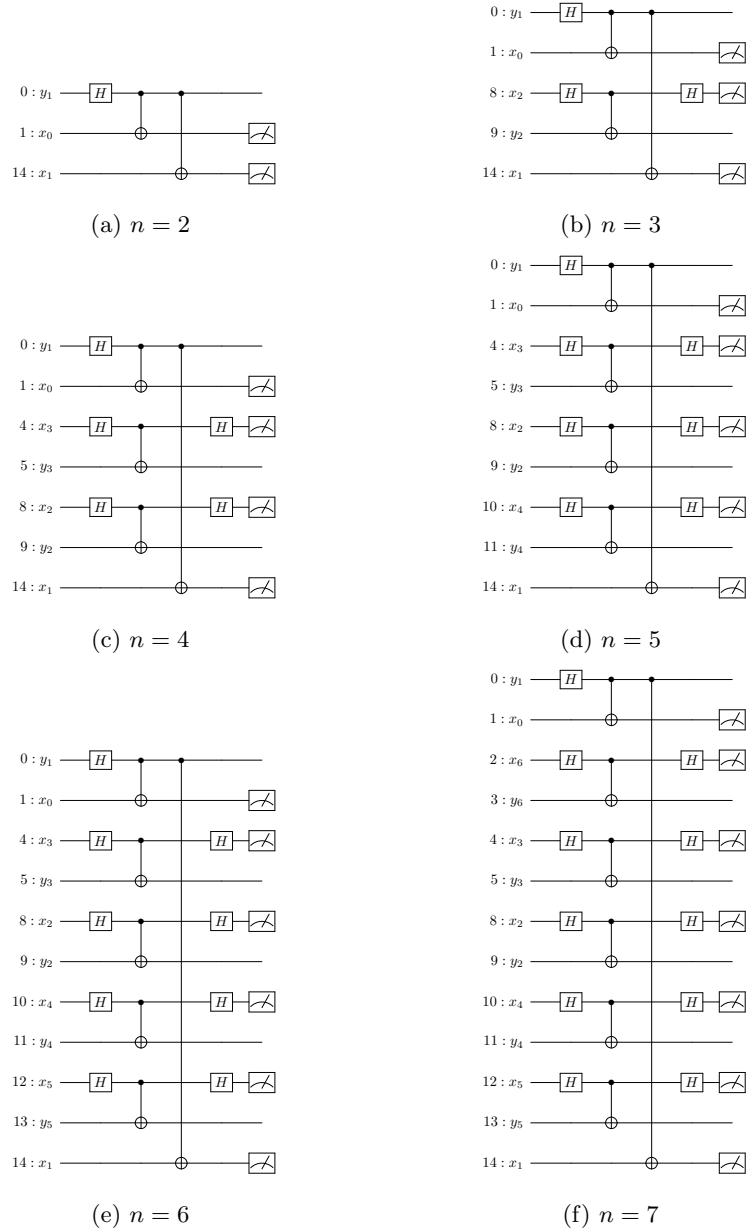


Fig. 17: Optimized circuits for $n = 2, \dots, 7$ with $\mathbf{s} = 0^{n-2}11$. We omit qubits 6, 7 and so input y_0 , which is not required after optimization.