Practical Period Finding on IBM Q – Quantum Speedups in the Presence of Errors

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. We implemented Simon’s quantum period finding circuit for functions $F_2^n \rightarrow F_2^n$ with period $s \in F_2^n$ up to $n = 7$ on the 14-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 $s$. While Simon’s algorithm for extracting $s$ runs in polynomial time in the error-free case $\tau(n) = 0$, we show that the problem of extracting $s \in F_2^n$ in the general setting $0 \leq \tau(n) \leq \frac{1}{2}$ is as hard as solving LPN (Learning Parity with Noise) with parameters $n$ and $\tau(n)$. Hence, in the error-prone case we may not hope to find periods in time polynomial in $n$. However, we also demonstrate theoretically and experimentally that erroneous quantum measurements are still useful to find periods faster than with purely classical algorithms, even for large errors $\tau(n)$ close to $\frac{1}{2}$.

Keywords: IBM Q 16, LPN, period finding, quantum supremacy

1 Introduction

The discovery of Shor’s quantum algorithm [18] 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) [5] and Learning with Errors (LWE) [16].

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 be easily handled by doubling the key size. However, starting with the initial work of Kuwakado, Morii [14] and followed by Kaplan, Leurent, Leverrier and Naya-Plasencia [13] 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 [19]. This is especially interesting, because Even and Mansour [11] proved that in the ideal cipher model any classical attack on their construction with $n$-bit keys requires $\Omega(2^n)$ steps.

∗ Funded by DFG under Germany’s Excellence Strategy - EXC 2092 CASA - 390781972.
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. [17,16,26,12,8,7]. In a nutshell, Simon’s quantum circuit produces for a periodic function $f : F_2^n \rightarrow F_2^n$ with period $s \in F_2^n$, i.e. $f(x) = f(z)$ iff $z \in \{x, x + s\}$, via quantum measurements uniformly distributed vectors $y$ that are orthogonal to $s$. It is not hard to see that from a basis of $y$’s that spans the subspace orthogonal to $s$, the period $s$ 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 therefore calls to $f$), and some polynomial time classical post-processing. On any purely classical computer however, finding the period of $f$ is equivalent to collision finding and thus requires $\Omega(2^n)$ operations.

Our contributions. We implemented Simon’s algorithm on IBM’s freely available Q 16 Melbourne [1], called IBM-Q16 in the following, that realizes 14-qubit quantum circuits. Since Simon’s quantum circuit requires for $n$-bit periodic functions $2^n$ qubits, we were able to implement functions up to $n = 7$ bits. Due to its limited size, IBM-Q16 is not capable of performing any error correction [9] on the circuits.

Implementation. Our experiments show that with some (significant) error probability $\tau$, we measure on IBM-Q16 vectors $y$ that are not orthogonal to $s$. The error probability $\tau$ 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 $\tau$. Since increasing $n$ requires an increasing amount of gates, we discovered experimentally that $\tau(n)$ increases as a function of $n$. For the function $f$ that we implemented, we found $\tau$-values ranging between $\tau(2) = 0.1$ and $\tau(7) = 0.15$.

Although IBM-Q16 produces faults for Simon’s quantum circuit, we still observe qualitatively the desired quantum effect: Vectors $y$ orthogonal to $s$ appear with significant larger probabilities than vectors not orthogonal to $s$. Moreover, experimentally our distribution among those vectors that are orthogonal (respectively not orthogonal) to $s$ is close to uniform. Notice that intuitively it should be hard to distinguish orthogonal vectors from not orthogonal ones.

Hardness. Based on our IBM-Q16 experiments, we obtain a (simplified) error model that any quantum measurement yields with probability $1 - \tau$ a uniformly chosen vector $y$ orthogonal to $s$, and with probability $\tau$ a uniformly chosen vector $y$ not orthogonal to $s$. We call Learning Simon with Noise (LSN) the problem of recovering $s \in F_2^n$ from quantum measurements. We show that solving LSN with parameters $n, \tau$ is polynomial time equivalent to solving the famous Learning Parity with Noise (LPN) problem with the same parameters $n, \tau$. The core of the reduction shows that LSN samples coming from quantum measurements of Simon’s circuit can be turned into perfectly distributed LPN samples, and vice versa.
Hence, quantum measurements of Simon’s circuit realize a physical LPN oracle. To the best of our knowledge, this is the first known physical realization of such an oracle. Moreover, from our hardness result we obtain a quite surprising link between symmetric and public key cryptography: Handling errors (i.e. not orthogonal vectors) in Simon’s algorithm, the most important quantum algorithm in symmetric crypto, is as hard as LPN, one of the major problems in post-quantum public key crypto.

From a cryptanalyst’s perspective, this result may at first sound quite negative, since we believe that we cannot solve LPN (and thus by the LPN-to-LSN reduction also LSN) in time polynomial in \((n, \tau)\) — not even on a quantum computer. On the positive side, the LSN-to-LPN reduction accurately tells us how harmful errors \(\tau\) from quantum computers are in practice, and how they affect the time complexity for quantum-assisted period finding.

**Error Handling.** We may use the LSN-to-LPN reduction to handle errors from IBM-Q16 via LPN-solving algorithms. In theory, the best algorithm for solving LPN with constant \(\tau\) is the BKW-algorithm of Blum, Kalai and Wasserman \([4]\) with time complexity \(2^{O\left(\frac{n}{\log(n\tau)}\right)}\). This already improves on the classical time \(2^\tau\) for period finding. However, the BKW-algorithm has a huge sample and memory complexity, which hinder its practical implementation.

At the moment, the largest LPN instances with errors in IBM-Q16’s range \(\tau \in [0.1, 0.15]\) are solved with variants of the low-memory algorithms POOLED GAUSS and WELL-POOLED GAUSS of Esser, Kübler, May \([10]\). 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 \(\tau\). For \(\tau = 0\), we obtain polynomial time as predicted by Simon’s analysis. However, for \(0 < \tau < \frac{1}{2}\) we achieve exponential run time, but still improve over the purely classical computation. This indicates that we achieve quantum supremacy for the 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 (overly optimistic) error-free Simon setting \(\tau = 0\), we obtain in the (realistic) general error-prone Simon setting \(0 < \tau < \frac{1}{2}\) only a polynomial speedup with a polynomial of degree \(\frac{1}{2c} > 1\).

Concerning quantum supremacy, assume that one could build a quantum device with 486 qubits performing Simon’s circuit on a 243-bit periodic function with error \(\tau = \frac{1}{8}\). Then the error handling would translate into an LPN-instance with \((n, \tau) = (243, \frac{1}{8})\). Such an LPN instance was solved in \([10]\) on 64 threads in only 15 days, whereas classically we would need \(2^{121}\) steps for period finding.

The paper is organized as follows. In Section 2 we recall Simon’s original quantum circuit, and already introduce our error model that we experimentally
verify in Section 3 on IBM-Q16. In Section 4 we show the polynomial time equivalence of LSN and LPN. In Section 5 we theoretically show that quantum measurements with error $\tau$ in combination with LPN-solvers outperform classical period finding for any $\tau < \frac{1}{2}$. Eventually, in Section 6 we experimentally extract periods out of erroneous IBM-Q16 measurements.

2 Simon’s Algorithm in the Presence of Errors

Notation. All logs in this paper are base 2. Let $x \in \mathbb{F}_2^n$ denote a binary vector with coordinates $x = (x_{n-1}, \ldots, x_0)$. Let $0 \in \mathbb{F}_2^n$ 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 $\text{Ber}_\tau$ the Bernoulli distribution for $\mathbb{F}_2$, i.e. a $0, 1$-valued $X \sim \text{Ber}_\tau$ satisfies $\Pr[X = 1] = \tau$.

Two vectors $x, y$ are orthogonal if their inner product $\langle x, y \rangle := \sum_{i=0}^{n-1} x_i y_i \mod 2$ is 0, otherwise they are called not orthogonal. Let $s \in \mathbb{F}_2^n$. Then we denote the subspace of all vectors orthogonal to $s$ as $s^\perp = \{x \in \mathbb{F}_2^n \mid \langle x, s \rangle = 0\}$.

Let $Y = \{y_1, \ldots, y_k\} \subseteq \mathbb{F}_2^n$. Then we define $Y^\perp = \{x \mid \langle x, y_i \rangle = 0 \text{ for all } i\}$.

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

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

Notice that $U_f(U_f(x, y)) = (x, y)$.

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 $H \otimes \ldots \otimes H$. Let $|x\rangle |y\rangle \in \mathbb{C}^4$ be a 2-qubit system. The $\text{cnot}$ (controlled $\text{not}$) 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 $\text{not}$ 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 $s \in \mathbb{F}_2^n \setminus \{0\}$ such that for all $x, y \in \mathbb{F}_2^n$ we have

$$f(x) = f(y) \iff y = x + s.$$ 

In Simon’s problem we have to find $s$ given oracle access to $f$.

In order to solve Simon’s problem classically, we have to find some collision $x \neq y$ satisfying $f(x) = f(y)$. It is well-known that this requires $\Omega(2^{\frac{n}{2}})$ function evaluations.
Simon’s quantum algorithm \cite{Simon}, called SIMON (see Algorithm 1), solves Simon’s problem with only $O(n)$ 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 $y \in \mathbb{F}_2^n$ that is orthogonal to $s$. Moreover, $y \in \mathbb{F}_2^n$ is uniformly distributed in the subspace $s^\perp$, t.i. we obtain each $y \in s^\perp$ with probability $\frac{1}{2^{n-1}}$. SIMON repeats to measure $Q_f^{SIMON}$ until it has collected $n-1$ linearly independent vectors $y_1, \ldots, y_{n-1}$, from which $s$ can be computed via linear algebra in polynomial time. It is not hard to see that a collection of $n-1$ linearly independent vectors requires only $O(n)$ function evaluations.

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{fig1.png}
\caption{Quantum circuit $Q_f^{SIMON}$}
\end{figure}

\begin{algorithm}
\caption{SIMON}
\begin{algorithmic}
\State \textbf{Input} : Simon function $f_s : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n$.
\State \textbf{Output}: Period $s \in \mathbb{F}_2^n$.
\State 1 Set $Y = \emptyset$.
\Repeat
\State 3 Run $Q_f^{SIMON}$ on $|0^n\rangle \otimes |0^n\rangle$.
\State 4 Let $y \in \mathbb{F}_2^n$ be the measurement of the first $n$ qubits.
\State 5 If $y \notin \text{span}(Y)$, then include $y$ in $Y$.
\Until $|Y| = n-1$
\State 7 Compute the unique $s \in Y^\perp \setminus \{0\}$.
\State 8 \Return $s$.
\end{algorithmic}
\end{algorithm}

At this point we should stress that SIMON only works for error-free quantum computations. Hence we have to ensure that each $y$ is indeed in $s^\perp$. Assume that we obtain in line 4 of algorithm SIMON at least a single $y$ with $\langle y, s \rangle = 1$. Then the output of SIMON is always false! Thus, SIMON is not robust with respect to computational errors on the quantum device.

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

\textbf{Definition 2.2 (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 $y \in s^\perp$, and with probability $\tau$ some uniformly random $y \in \mathbb{F}_2^n \setminus s^\perp$. That is, the output distribution is

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

We call $\tau$ the error rate of our quantum device.

In the subsequent Section 3 we show that the IBM-Q16 realization of quantum circuits approximately follows our error model of Definition 2.2.

Notice that intuitively there is no efficient way to tell whether $y \in s^\perp$. This intuition is stated more precisely in Section 4, where we show that computing $s$ from the distribution in Definition 2.2 is 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, which (despite its name) realizes 14-qubit circuits. Let us number IBM-Q16’s qubits as $0, \ldots, 13$. Our implementation goal was to realize quantum period finding for Simon functions $f_s : \mathbb{F}_2^n \to \mathbb{F}_2^n$ 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 $\text{cnot}$. On IBM-Q16, the application of any gates introduces some error, where especially the 2-qubit $\text{cnot}$ introduces approximately as much error as ten 1-qubit gates (see Appendix A, Table 3). 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 directed topology graph, where node $i$ denotes qubit $i$. Moreover, let $\bar{G} = (V, \bar{E})$ be the undirected version of $G$, i.e. we have $\{u, v\} \in \bar{E}$ iff $(u, v) \in E$ or $(v, u) \in E$.

If $(u, v) \in E$ then we can directly implement $\text{cnot}(u, v)$, where $u$ serves as the control bit. If we wish to implement $\text{cnot}(v, u)$ instead, we may use the identity of Figure 3 at the cost of an additional 4 Hadamard gates. Hence, we call qubits $u, v$ adjacent iff $\{u, v\} \in \bar{E}$.

Let us assume that we want to realize $\text{cnot}(1, 3)$ in our algorithm. Since $\{1, 3\} \not\in \bar{E}$ we cannot directly realize this operation. But we may first swap the contents of qubits 2 and 3 by realizing a $\text{swap}$ gate via 3 $\text{cnot}$s as depicted in Figure 4. Since $(2, 3) \in E$, we realize the first and third $\text{cnot}$ directly, whereas the second $\text{cnot}$ is realized as in Figure 3. Thus, with a total of 3 $\text{cnot}$ and 4 Hadamards we swap the content of qubit 3 into 2. Since $(1, 2) \in E$, we may now apply $\text{cnot}(1, 2)$. 

6
Fig. 2: Topology graph $G(V,E)$ of IBM-Q16.

$$G(V,E)$$

Fig. 3: Control bit change

Function choice. Let $s \in \mathbb{F}_2^n \setminus \{0\}$, and let $i \in [0,n-1]$ with $s_i = 1$. We define

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

Let us first show that $f_s$ is indeed a Simon function as given in Definition 2.1. We have for all $x \in \mathbb{F}_2^n$ that

$$f_s(x + s) = x + s + (x + s)_i \cdot s = x + s + (x_i + 1) \cdot s = x + x_i \cdot s = f_s(x).$$

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

$$x + x_i \cdot s = y + y_i \cdot s.$$  

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

Fig. 4: Realisation of swap via 3 cnots and 4 Hadamards.
Instantiation and Discussion of Function Choice. Throughout the paper, we instantiate our function $f_s$ with the period $s = (s_{n-1}, \ldots, s_0) = 0^{n-2}11$ and $x_i = x_0$. We may realize $f_s$ with $n$ cnot-gates for copying $x$, and an additional 2 cnot-gates for the controlled addition of $s$ via control bit 0. See Figure 5 for an implementation of $f_s$ with $n = 3$.

![Simon circuit Q1](image)

Fig. 5: Simon circuit $Q_1$ with our realization of $f_s$ and $\text{CN}(Q_1) = 56$. The first 3 cnots copy $x$, the remaining two cnots add $s = 110$.

Our function choice has the advantage that it can be implemented with only $n + 2$ cnot gates (if we are able to avoid swaps) and $2n$ Hadamards. 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 3.1.

We would like to point out that as a downside of its simplicity, for our class of functions $f_s$ it is classically not hard to find the period $s$. Since $f(1^n) + 1^n = s$, a single classical $f$-query directly reveals $s$. However, we want to stress that our quantum algorithm does not exploit this property of $f$ in any manner, but instead works for any Simon function. The only reason that we use our simple form $f$ is that IBM-Q16 forces us to have a low circuit norm for producing a tolerable error.

3.1 Minimizing the gate count of $f_s$

We may implement $f_s$ on IBM-Q16 directly as the circuit $Q_1$ from Figure 5. Since $Q_1$ uses 6 Hadamard- and 5 cnot-gates, we have circuit norm $\text{CN}(Q_1) = 56$, but only when ignoring IBM-Q16’s topology. As already discussed, IBM-Q16 only allows cnots between adjacent qubits in the topology graph $G = (V, E)$ of Figure 2.

Thus, IBM-Q16 compiles $Q_1$ to $Q_2$ as depicted in Figure 6. Let us check that $Q_2$ realizes the same circuit as $Q_1$, but only acts on adjacent qubits. Let $U_{f_s} : \mathbb{F}_2^5 \rightarrow \mathbb{F}_2^5$ be the universal quantum embedding of $f_s$ with $(x, y) \mapsto (x, f(x) + y) =$
x + x_0 s + y \). In \( U_{f_s} \) we first add each \( x_i \) to \( y_i \) via \texttt{cnot}s, see Figure 5. Thus, we have to make sure that each \( x_i \) is adjacent to its \( y_i \). Second, we add \( s = 011 \) via \texttt{cnot}s controlled by \( x_0 \). Thus, we have to ensure that \( x_0 \) is adjacent to \( y_0 \) and \( y_1 \).

We denote by \( i : j \) that qubit \( i \) contains the value \( j \). This allows us to define the starting configuration as

\[
0 : x_0 \quad 1 : x_1 \quad 2 : x_2 \quad 3 : y_0 \quad 4 : y_1 \quad 5 : y_2.
\]

Step 1 of \( Q_2 \) (see Figure 5) performs \texttt{swap}(2, 3) and thus results in configuration

\[
0 : x_0 \quad 1 : x_1 \quad 2 : y_0 \quad 3 : x_2 \quad 4 : y_1 \quad 5 : y_2.
\]

Step 2 of \( C_2 \) performs \texttt{swap}(1, 2) as well as \texttt{swap}(4, 3). This results in configuration

\[
0 : x_0 \quad 1 : y_0 \quad 2 : x_1 \quad 3 : y_1 \quad 4 : x_2 \quad 5 : y_2.
\]

Eventually, Step 3 of \( C_2 \) performs \texttt{swap}(0, 1) and \texttt{swap}(2, 3) resulting in

\[
0 : y_0 \quad 1 : x_0 \quad 2 : y_1 \quad 3 : x_1 \quad 4 : x_2 \quad 5 : y_2.
\]

Since \( (1, 0), (2, 3), (5, 4) \in E \), in Step 4 we now compute \texttt{cnot}(1, 0), \texttt{cnot}(3, 2) and \texttt{cnot}(4, 5) by changing for the second and third operation the control bit (see Figure 3). This realizes the computation of \( x + y \). For realizing the addition of \( x_i \cdot s = x_0 \cdot 011 \), in Step 5 we compute \texttt{cnot}(1, 0) and \texttt{cnot}(1, 2) using \( (1, 0), (1, 2) \in E \).

\[
\begin{array}{c}
0 : x_0 \\
1 : x_1 \\
2 : x_2 \\
3 : y_0 \\
4 : y_1 \\
5 : y_2
\end{array}
\]

Fig. 6: IBM-Q16 compiles \( Q_1 \) to \( Q_2 \) with \( \text{CN}(Q_2) = 234 \).

In total \( Q_2 \) consumes 34 1-bit gates and 20 2-bit gates and thus has \( \text{CN}(Q_2) = 234 \), as compared to \( \text{CN}(Q_1) = 56 \). In the following, our goal is the construction of a quantum circuit that implements \( Q_1 \)'s functionality with minimal circuit norm on IBM-Q16.

In Figure 7 we start with circuit \( Q_3 \), for which our optimization eventually results in circuit \( Q_4 \) (Figure 9) that can be realized on IBM-Q16 with gate count only \( \text{CN}(Q_4) = 33 \).
Fig. 7: Circuit $Q_3$.

From the discussion before, it should not be hard to see that $Q_3$ realizes $Q_{\text{Simon}}$, but yet it has to be optimized for IBM-Q16. First of all observe that \texttt{cnot} is self-inverse, and thus we can eliminate the two \texttt{cnot}(2,3) gates. Afterwards, we can safely remove qubit 3. The resulting situation for qubits 0, 1, 2 is depicted in Figure 8.

Fig. 8: Optimization of $Q_3$.

From Figure 8 we see that the change of control bits from $\texttt{cnot}(0,1), \texttt{cnot}(2,1)$ to $\texttt{cnot}(1,0), \texttt{cnot}(1,2)$ 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 $\text{CN}(Q_4) = 33$.

Since $(1,0), (1,2), (6,8) \in E$, all three \texttt{cnot}s 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
error rate (see Figure 2). The choice of our configurations is given in Table 1, a complete list of optimized circuits can be found in Appendix A, Figure 11.

### Table 1: Table of configurations.

| $n$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | CN |
|-----|---|---|---|---|---|---|---|---|---|---|----|----|----|----|----|
| 2   |  |   |   |   |   |   |   |   |   |   |    |    |    |    | 21 |
| 3   |   |   |   |   |   |   |   |   |   |   |    |    |    |    | 33 |
| 4   |   |   |   |   |   |   |   |   |   |   |    |    |    |    | 45 |
| 5   |   |   |   |   |   |   |   |   |   |   |    |    |    |    | 57 |
| 6   |   |   |   |   |   |   |   |   |   |   |    |    |    |    | 69 |
| 7   |   |   |   |   |   |   |   |   |   |   |    |    |    |    | 81 |

### 3.2 Experiments on IBM Q 16

For each dimension $n = 2, \ldots, 7$ we took 8192 measurements on IBM-Q16 of our optimized circuits from the previous section. The resulting relative frequencies are depicted in Figure 10. For each $n$, let $S(n)$ denote the set of erroneous measurements in $\mathbb{F}_2^n \setminus s^\perp$. Then we compute the error rate $\tau(n)$ as $\tau(n) = \frac{|S(n)|}{8192}$.

In Figure 10, we draw horizontal lines $\frac{1-\tau(n)}{2^{n-1}}$, respectively $\frac{\tau(n)}{2^{n-1}}$, for the probability distributions of our error model for orthogonal, respectively not orthogonal, vectors.

We observe the following:

- **Vectors in $s^\perp$ are more frequent.** We see that in principle IBM-Q16 works well for period finding. E.g. for $n = 3$, we have $\{s\}^\perp = \{011\}^\perp = \{000, 011, 100, 111\}$, and we measure one of these vectors with probability $1 - \tau \approx 90\%$. This in turn implies that we also measure erroneous vectors in $\mathbb{F}_2^n \setminus \{s\}^\perp$ with probability $\tau \approx 10\%$. 

![Fig. 9: Optimized circuit $Q_4$ on IBM-Q16 with CN($Q_4$) = 33.](image-url)
Fig. 10: IBM-Q16 measurements of our optimized circuits (see Appendix [A Figure 11]).

- **Triangular structure.** In Figure 10 we ordered our measurements lexicographically. It seems that the first vectors in lexicographic order are measured with a larger probability than the last vectors. This is not overly surprising,
since we also ordered our qubits by quality – starting with 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 \(n = 3\) it seems that the qubit for \(x_2\) performed
worse than the one for \(x_3\)). We can mitigate the effect of different qubit qual-
yty by permuting the qubits in our starting configuration such that we retain
the same circuit norm. However, we deliberately ordered the qubits in de-
sceding quality to make the quality effect visible. The triangular structure
would vanish for qubits of similar quality.

- 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.

- Hamming weight. Usually, measurements with small Hamming weight
appear with larger frequencies than large Hamming weight measurements.
This is a physical effect that is mainly due to the readout error of the mea-
surements in IBM-Q16 (see Appendix A, Table 3) and its significant bias
towards 0.

All in all, our error model is an oversimplified model that for ease of exposi-
tion ignores facts like error quality of different qubits and issues with Hamming
weight. It is not surprising that a single parameter like the error rate \(\tau\) cannot
all too precisely capture complex physical effects and complex probability dis-
tributions. Nevertheless, we show in Section 6 that our simple error model is
accurate enough to predict the run times for extracting the secret vector \(s\) from
quantum measures with error rate \(\tau\).

### 4 LSN is Polynomial Time Equivalent to LSN

In the previous section, we checked experimentally on IBM-Q16 our error model
(Definition 2.2). Recall that our model states that with probability \(\tau\) we measure
in the quantum circuit \(Q_{\text{Simon}}\) some uniformly distributed \(y \in \mathbb{F}_2^n \setminus s^\perp\). The
question is now whether such erroneous \(y\) can easily be handled.

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

**Definition 4.1 (LPN-Problem).** Let \(s \in \mathbb{F}_2^n \setminus \{0\}\) be chosen uniformly at
random, and let \(\tau \in [0, \frac{1}{2})\). In the Learning Parity with Noise problem, de-
noted \(\text{LPN}_{n,\tau}\), one obtains access to an oracle \(O_{\text{LPN}}(s)\) that provides samples
\((a, \langle a, s \rangle + \epsilon)\), where \(a \sim \mathcal{U}_n\) and \(\epsilon \sim \text{Ber}_\tau\). The goal is to compute \(s\).

Definition 4.1 explicitly excludes \(s = 0\) in LPN. Notice that the case \(s = 0\)
implies that the LPN oracle has distribution \(U_n \times \text{Ber}_\tau\). However, in the case
\(s \neq 0\), we have \(P_a[\langle a, s \rangle = 0] = \frac{1}{2}\), which implies \(P_a[\langle a, s \rangle + \epsilon = 0] = \frac{1}{2}\). Therefore
the LPN samples for \(s \neq 0\) have distribution \(U_n \times U\). This allows us to easily
distinguish both cases by a majority test, whenever \(\tau\) is polynomially bounded
away from \(\frac{1}{2}\). Hence, \(s = 0\) is not a hard case for LPN and may wlog excluded.
Let us now define the related Learning Simon with Noise problem that reflects our error model.

**Definition 4.2 (LSN-Problem).** Let \( s \in \mathbb{F}_2^n \setminus \{0\} \) 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 \( \mathcal{O}_{\text{LSN}}(s) \) that provides samples \( y \), where \( y \in \mathbb{F}_2^n \) is distributed as in Definition 2.2, i.e.

\[
\mathbb{P}[y] = \begin{cases} 
\frac{1-\tau}{2^n}, & \text{if } y \in s^\perp \\
\frac{\tau}{2^n}, & \text{else}
\end{cases}
\]

and therefore \( \mathbb{P}[\langle y, s \rangle = 0] = 1 - \tau \).

The goal is to compute \( s \).

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 \( \mathcal{O}_{\text{LPN}}(s) \) and \( \mathcal{O}_{\text{LSN}}(s) \).

**Theorem 4.1 (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 \( \epsilon_A \). 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 \( \epsilon_B \geq \frac{\epsilon_A^2}{4} \).

**Proof.** Assume that we want to solve LSN via an algorithm \( \mathcal{A}_{\text{LPN}} \) with success probability \( \epsilon_A \) as in Algorithm 2.

**Algorithm 2: LPN \( \Rightarrow \) LSN**

| Input       | : \( n, \tau, \mathcal{O}_{\text{LSN}}(s), m \) |
|-------------|------------------------------------------------|
| Output      | : \( s \)                                     |
| 1           | Choose \( z \sim \mathcal{U}_n \).            |
| 2 for i = 1 to m do |
| 3           | Set \( y_i \sim \mathcal{O}_{\text{LSN}}(s) \).|
| 4           | Choose \( b_i \sim \mathcal{U}_n \).          |
| 5 end       |                                               |
| 6 s \leftarrow \mathcal{A}_{\text{LPN}}(n, \tau, \langle y_1 + b_1 z, b_1 \rangle, \ldots, \langle y_m + b_m z, b_m \rangle) |}

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

\[
\epsilon_B \geq \mathbb{P}_s[\langle z, s \rangle = 1 \cap \mathcal{A} \text{ outputs } s] = \frac{\epsilon_A}{2}.
\]

Let us now show correctness of Algorithm 2 We first show that the constructed LPN samples \( (y + bz, b) \) have the correct distribution. Let \( \epsilon = (y + b z, b) \)
Since \( (z, s) = 1 \), we have
\[
P_y[\epsilon = 1] = P_y[(y + bz, s) + b = 1] = P_y[(y, s) + b(z, s) + b = 1] = P_y[(y, s) = 1] = \tau.
\]

It remains to show that \( y + bz \) is uniformly distributed. To this end, we show that
\[
p_0 = P_{y,b}[y + bz \mid \langle y, s \rangle = 0] = \frac{1}{2^n}.
\]

Analogously, it follows that
\[
p_1 = P_{y,b}[y + bz \mid \langle y, s \rangle = 1] = \frac{1}{2^n}.\]

From both statements we obtain
\[
P_{y,b}[y + bz] = P_y[(y, s) = 0] \cdot p_0 + P_y[(y, s) = 1] \cdot p_1 = \frac{1 - \tau}{2^n} + \frac{\tau}{2^n} = \frac{1}{2^n},
\]

as desired. It remains to show that
\[
p_0 = P_{y,b}[y + bz \mid \langle y, s \rangle = 0] = \frac{1}{2^n} \left( \frac{1 - \tau}{2^n} + \frac{\tau}{2^n} \right) = \frac{1}{2^n}.
\]

This completes the analysis of Algorithm 2.

---

**Algorithm 3: LSN \(\Rightarrow\) LPN**

**Input:** \( n, \tau, \mathcal{O}_{LPN}(s), m \)

**Output:** \( s \)

1. Choose \( z \sim U_n \).
2. For \( i = 1 \) to \( m \) do
3. \hspace{1em} Set \( (a_i, b_i) \leftarrow \mathcal{O}_{LPN}(s) \).
4. End
5. \( s \leftarrow A_{LSN}(n, \tau, a_1 + b_1 z, \ldots, a_m + b_m z) \)

For Algorithm 3, we conclude the success probability analogous to the reasoning for Algorithm 2, i.e., we succeed when \( (z, s) = 1 \) and \( A_{LSN} \) succeeds. So let us assume in the following correctness analysis that we are in the case \( (z, s) = 1 \). This implies for the constructed LSN samples \( a + bz \) that
\[
\langle a + bz, s \rangle = 0 \iff \langle a, s \rangle + b(z, s) = 0 \iff \langle a, s \rangle = b.
\]

Let \( \epsilon = \langle a, s \rangle + b \). It follows that
\[
P_{a,b}[\langle a + bz, s \rangle = 0] = P_{a,b}[\langle a, s \rangle = b] = P_{a,b}[\epsilon = 0] = 1 - \tau.
\]

We also have to show that we obtain a uniform distribution among all \( a + bz \in s^\perp \).
This follows from
\[
P_{a,b}[a + bz \mid \langle a + bz, s \rangle = 0] = P_{a,b}[a + bz \mid \langle a, s \rangle = b]
\]

15
\[ P_a[(a, s) = 0] \cdot P_{a,b}[a + b z \mid \langle a, s \rangle = b = 0] + \]
\[ P_a[(a, s) = 1] \cdot P_{a,b}[a + b z \mid \langle a, s \rangle = b = 1] \]
\[ = \frac{1}{2} \cdot P_a[a \mid \langle a, s \rangle = 0] + \frac{1}{2} \cdot P_a[a + z \mid \langle a, s \rangle = 1] \]
\[ = \frac{1}{2} \cdot \frac{1}{2^{n-1}} + \frac{1}{2} \cdot \frac{1}{2^{n-1}} = \frac{1}{2^{n-1}}. \]

Analogously, we can show that we obtain a uniform distribution among all \( a + b z \in \mathbb{F}_2^n \setminus s^\perp \). This proves that we perfectly simulate LSN-samples via \( O_{\text{LPN}} \), and thus shows correctness of Algorithm 3.

Theorem 4.1 shows that under the LPN assumption we cannot expect to solve LSN — i.e. to handle error-prone quantum measurements in Simon’s algorithm — in polynomial time. However, it does not exclude that quantum measurements 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 our quantum output indeed leads to speedups even for large error rates \( \tau \).

5 Theoretical Error Handling for Simon’s Algorithm

Recall that period finding for \( n \)-bit Simon functions classically requires time \( \Omega(2^n) \). So despite the hardness results of Section 4 we may still hope that even error-prone quantum measurements lead to period finding speedups. Indeed, it is well-known that for any fixed \( \tau < \frac{1}{2} \) the BKW algorithm [4] solves LPN\(_n,\tau \) — and thus by Theorem 4.1 also LSN\(_n,\tau \) — in time \( 2^{O \left( \frac{n^2}{\log n} \right)} \). This implies that asymptotically the combination of quantum measurements together with a suitable LPN-solver already outperforms classical period finding.

However, the BKW algorithm has sample and memory consumption \( 2^{\Theta \left( \frac{n^2}{\log n} \right)} \), which makes it quite impractical in practice. Therefore, we want to focus on LPN-algorithms that consume only a small amount of samples and memory. We start with the analysis of the POOLED GAUSS algorithm that was introduced at Crypto ’17 by Esser, Kübler and May [10]. POOLED GAUSS solves LPN\(_n,\tau \) in time \( \tilde{O} \left( 2^{\log \left( \frac{1}{1-\tau} \right) n} \right) \) using \( \tilde{O} \left( n^2 \right) \) samples and \( \tilde{O} \left( n^3 \right) \) memory.

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 \( \tau \) is bounded by \( \tau \leq 0.293 \).

**Theorem 5.1.** In our error model (Definition 2.2), POOLED GAUSS finds the period \( s \in \mathbb{F}_2^n \) of a Simon function \( f_s \) using \( \tilde{O} \left( n^2 \right) \) many LSN\(_n,\tau\)-samples, coming from practical measurements of Simon’s circuit \( Q_{f_s}^{\text{SIMON}} \) with error rate \( \tau \), in time \( \tilde{O} \left( 2^{\log \left( \frac{1}{1-\tau} \right) n} \right) \). This improves over classical period finding for error rates

\[ \tau < 1 - \frac{1}{\sqrt{2}} \approx 0.293. \]
Proof. We use Algorithm 2 where any $O_{\text{LPN}}(s)$-call is provided by a measurement of $Q_{s}^{\text{Simon}}$. In our error model, this gives us an LSN$_{n,\tau}$-instance which is transformed by Algorithm 2 into an LPN$_{n,\tau}$-instance. We use POOLED GAUSS as the LPN-solver $A_{\text{LPN}}$ inside Algorithm 2. This immediately implies time complexity $\tilde{\Theta}(2^{\log(\frac{1}{1-\tau})}n)$.

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 5.1 already shows the usefulness of a quite limited quantum oracle that only allows us polynomially many measurements, whenever its error rate $\tau$ is small enough.

If we allow for more quantum measurements, the WELL-POOLED GAUSS algorithm of Esser, Kübler and May [10] solves LPN$_{n,\tau}$ in improved time and query complexity $\tilde{\Theta}(2^{f(\tau)n})$, where $f(\tau) = 1 - \frac{1}{1+\log(\frac{1}{1-\tau})}$, using $\tilde{\Theta}(n^3)$ memory.

The following theorem shows that WELL-POOLED GAUSS in combination with error-prone quantum measurements improves on classical period finding for any error rate $\tau$.

**Theorem 5.2.** In our error model (Definition 2.2), WELL POOLED GAUSS finds the period $s \in \mathbb{F}_2^n$ of a Simon function $f_s$ using $\tilde{\Theta}(2^{f(\tau)n})$ many LSN$_{n,\tau}$ samples, coming from practical measurements of Simon’s circuit $Q_{f_s}^{\text{Simon}}$ with error rate $\tau$, 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 5.2 we use Algorithm 2 where measurements of $Q_{f_s}^{\text{Simon}}$ provide the $O_{\text{LPN}}(s)$-calls and WELL POOLED GAUSS is the LPN-solver $A_{\text{LPN}}$. 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 5.1 and Theorem 5.2 show that quantum measurements of $Q_{f_s}^{\text{Simon}}$ always help us (asymptotically) even for large error rates $\tau$, provided that our error model is sufficiently accurate. In the following section, we show that our simple error model is in practical experiments sufficiently precise to predict run times.
In this section, we want to handle errors in quantum measurements of Simon’s circuit $Q^{Simon}_{f_s}$ in practice. By the result of Section 4, we may first transform our quantum measurements into LPN samples, and then use one of the LPN-algorithms from Section 5. Since the error rates from our IBM-Q16 measurements in Section 3.2 are below error rate $\tau \leq 0.15$, according to Theorem 5.1 for sufficiently large $n$ LPN-solver POOLED GAUSS already outperforms classical period finding.

The goal of this section is to check the accuracy of our error model with respect to the prediction of run times in the presence of errors. Therefore, we do not implement the error handling detour via reduction to LPN, but we tackle the LSN problem directly. To this end, we simply adapt the POOLED GAUSS algorithm to the LSN setting. This is done in Algorithm 3, called POOLED SIMON.

**Algorithm 4: POOLED SIMON**

Input: Pool $P \subseteq \mathbb{F}_2^n$ of LSN samples with $|P| \geq n - 1$, oracle access to $f_s$

Output: Secret $s$.

1. begin
2. repeat
3. Randomly select linearly independent set $Y = \{y_1, \ldots, y_{n-1}\} \subseteq P$.
4. Compute the unique $s' \in Y^\perp \setminus \{0\}$
5. until $f_s(s') = f_s(0)$

Output: $s'$.

6. end

In practice, the input pool $P$ consists of $Q^{Simon}_{f_s}$-measurements. It is not hard to see that POOLED SIMON succeeds if $y_i \in s^\perp$ for all $y_i \in Y$. Thus, POOLED SIMON works similar to the original Simon algorithm (Algorithm 1), but repeats until it finds some error-free set $Y$ of measurements. If we would take fresh quantum measurements in each iteration of the repeat-loop, then we succeed in a single iteration with probability $(1 - \tau)^n$. This implies an expected run time of $(\frac{1}{1-\tau})^n$ iterations for POOLED SIMON. It was shown in [10] that this analysis also holds if we choose $Y \subseteq P$ for sufficiently large pools $P$.

In order to check the accuracy of our error model, we first ran POOLED SIMON for every $n = 2, \ldots, 7$ with our pools $P$ of $2^{13}$ quantum measurements from Section 3.2. Second, we also ran POOLED SIMON with pools $P$ of $2^{13}$ randomly chosen, perfectly distributed LSN samples. As the run time cost we took the number of $f_s$ evaluations, i.e. the number of iterations plus one for evaluating $f_s(0)$, averaged over 1000 runs of POOLED SIMON. In Table 2, we give the resulting run times $T_{QM}$ for our quantum measurements and $T_{LSN}$ for LSN samples.

From Table 2, we see that $T_{LSN}$ quite accurately predicts $T_{QM}$, but that in general it slightly underestimates $T_{QM}$. This in turn implies that for POOLED
Table 2: POOLED SIMON run times $T_{QM}$ for quantum measurements and $T_{LSN}$ for LSN samples.

| $(n, \tau(n))$ | $T_{QM}$ | $T_{LSN}$ | $\frac{T_{QM} - T_{LSN}}{n^2}$ |
|----------------|---------|---------|-------------------------------|
| (2, 0.098)     | 1.238   | 1.216   | 5.5 \cdot 10^{-3}            |
| (3, 0.099)     | 1.443   | 1.398   | 5.0 \cdot 10^{-3}            |
| (4, 0.109)     | 1.727   | 1.644   | 5.2 \cdot 10^{-3}            |
| (5, 0.126)     | 2.160   | 2.066   | 3.8 \cdot 10^{-3}            |
| (6, 0.118)     | 2.366   | 2.245   | 3.4 \cdot 10^{-3}            |
| (7, 0.147)     | 3.440   | 3.251   | 3.9 \cdot 10^{-3}            |

SIMON it might be a bit harder to handle errors in quantum measurement than solving LSN (or equivalently LPN). This seems reasonable because POOLED SIMON should profit from the LSN samples’ uniformity. However, this does not exclude other algorithms that might be tailored to and profit from the specific distribution of quantum measurements.

But of course, we should not over-interpret our very small run times in very small dimension. Assume e.g. that $T_{QM}$ and $T_{LSN}$ differ only by an additive term $O(n^2)$. Then the term $\frac{T_{QM} - T_{LSN}}{n^2}$ should be upper bounded by a constant, which seems to hold quite well for the limited data in Table 2. In this case, our error model would be (asymptotically) highly accurate. Only experiments on quantum devices with more qubits can tell us more.

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

1. 14-qubit backend: IBM Q team, "IBM Q 16 Melbourne backend specification V1.0.0," (2019). Retrieved from https://quantum-computing.ibm.com. Accessed 13. September 2019.
2. 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)
3. Alekhnovich, M.: More on average case vs approximation complexity. In: 44th FOCS, pp. 298–307. IEEE Computer Society Press (Oct 2003)
4. 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)
5. 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)
6. 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., 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)

11. 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)

12. 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)

13. 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)

14. 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/

15. 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)

16. 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)

17. 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

18. Shor, P.W.: Algorithms for quantum computation: Discrete logarithms and factoring. In: 35th FOCS. pp. 124–134. IEEE Computer Society Press (Nov 1994)

19. Simon, D.R.: On the power of quantum computation. In: 35th FOCS. pp. 116–123. IEEE Computer Society Press (Nov 1994)
A Appendix

| Parameter                        | Average measurement | Range of measurements       |
|----------------------------------|---------------------|-----------------------------|
| Gate error (Hadamard)            | $6.28 \cdot 10^{-3}$| $(1.67 \cdot 10^{-3}, 14.43 \cdot 10^{-3})$ |
| Gate error (cnot)                | $7.83 \cdot 10^{-2}$| $(3.15 \cdot 10^{-2}, 13.47 \cdot 10^{-2})$ |
| Readout error                    | $6.48 \cdot 10^{-2}$| $(2.58 \cdot 10^{-2}, 17.83 \cdot 10^{-2})$ |
| T1 ($\mu s$)                     | 50.54               | $(23.65, 91.36)$            |
| T2 ($\mu s$)                     | 69.14               | $(25.21, 119.98)$           |

Table 3: Calibration facts for our IBM-Q16 measurements
Fig. 11: Optimized circuits for $n = 2, \ldots, 7$ with $s = 0^{n-2}11$. We omit qubit 7 with input $y_0$, which is not required after optimization.