IQP Sampling and Verifiable Quantum Advantage: Stabilizer Scheme and Classical Security

Michael J. Bremner*[1], Bin Cheng†[1], and Zhengfeng Ji‡[2]

[1]*Centre for Quantum Computation and Communication Technology, Centre for Quantum Software and Information, School of Computer Science, Faculty of Engineering and Information Technology, University of Technology Sydney, NSW, Australia*

[2]*Department of Computer Science and Technology, Tsinghua University, Beijing, China*

Abstract

Sampling problems demonstrating beyond classical computing power with noisy intermediate scale quantum (NISQ) devices have been experimentally realized. In those realizations, however, our trust that the quantum devices faithfully solve the claimed sampling problems is usually limited to simulations of smaller-scale instances and is, therefore, indirect. The problem of verifiable quantum advantage aims to resolve this critical issue and provides us with greater confidence in a claimed advantage. Instantaneous quantum polynomial-time (IQP) sampling has been proposed to achieve beyond classical capabilities with a verifiable scheme based on quadratic-residue codes (QRC). Unfortunately, this verification scheme was recently broken by an attack proposed by Kahanamoku-Meyer.

In this work, we revive IQP-based verifiable quantum advantage by making two major contributions. Firstly, we introduce a family of IQP sampling protocols called the *stabilizer scheme*, which builds on results linking IQP circuits, the stabilizer formalism, coding theory, and an efficient characterization of IQP circuit correlation functions. This construction extends the scope of existing IQP-based schemes while maintaining their simplicity and verifiability. Secondly, we introduce the *Hidden Structured Code* (HSC) problem as a well-defined mathematical challenge that underlies the stabilizer scheme. To assess classical security, we explore a class of attacks based on secret extraction, including the Kahanamoku-Meyer's attack as a special case. We provide evidence of the security of the stabilizer scheme, assuming the hardness of the HSC problem. We also point out that the vulnerability observed in the original QRC scheme is primarily attributed to inappropriate parameter choices, which can be naturally rectified with proper parameter settings.

1 Introduction

Quantum computing represents a fundamental paradigm change in the theory of computation, and promises to achieve quantum speedup in many problems, such as integer factorization [1]

*Michael.Bremner@uts.edu.au
†Bin.Cheng@student.uts.edu.au
‡jizhengfeng@tsinghua.edu.cn

and database search [2]. However, many quantum algorithms are designed to be implemented in the fault-tolerant regime, which are too challenging for our current noisy intermediate-scale quantum (NISQ) era [3]. Experimentally, we can perform random-circuit sampling [4, 5, 6, 7] and boson sampling [8, 9] at a scale that is arguably beyond the capability of classical simulation. But when it comes to verifiability, although these experiments can use some benchmarking techniques such as cross-entropy benchmarking [5] to certify the quantum devices, they cannot be efficiently verified in an adversarial setting without modification of the underlying computational task.

Classical verification of quantum computation is a long-standing question, which was first asked by Gottesman [10]. In the context of verifying arbitrary quantum computation, there have been a plethora of important results [11, 12, 13, 14, 15, 16, 17, 18, 19]. The more relevant context of this work is generating a test of quantumness. The goal is to create a computational task that is beyond the capabilities of classical computing, that uses minimal quantum and classical computing to generate and verify. A motivating example is given by Shor's algorithm for integer factorization [1], which is appealing in that hard instances can be easily generated and verified classically yet finding the solution is beyond the capabilities of classical computers. However, this also has the drawback that the quantum solution also seems to be beyond the capabilities of NISQ devices.

Recently, there have been tests of quantumness that combine the power of both interactive proofs and cryptographic assumptions [20, 21, 22]. This class of cryptographic verification protocols usually uses a primitive called trapdoor claw-free (TCF) functions, which has the following properties. First, it is a 2-to-1 function that is hard to invert, meaning that given $y = f(x) = f(x')$, it is hard for an efficient classical computer to find the preimage pair (x, x'). Second, given a trapdoor to the function $f(x)$, the preimage pair can be efficiently found on a classical computer. We will refer to this class of verification protocols as the TCF-based protocols. The TCF-based protocols require the quantum prover to prepare the state of the form $\sum_x |x\rangle |f(x)\rangle$. Although a recent experiment implemented a small-scale TCF-based protocol on a trapped-ion platform [23], implementing this class of protocols is still very challenging for the current technology.

Another class of verification protocols is based on instantaneous quantum polynomial-time (IQP) circuits initiated by Shepherd and Bremner [24]. IQP circuits are a family of quantum circuits that employ only commuting gates, typically diagonal in the Pauli-X basis. In IQP-based verification protocols, the verifier generates a pair consisting of an IQP circuit U_{IQP} and a secret key $\mathbf{s} \in \{0,1\}^n$. After transmitting the classical description of the IQP circuit to the prover, the verifier requests measurement outcomes in the computational basis. Then, the verifier uses the secret to determine whether the measurement outcomes are from a real quantum computer. Such a challenge seems hard for classical computers, as random IQP circuits are believed to be computationally difficult to simulate classically with minimal physical resources, assuming some plausible complexity-theoretic assumptions such as the non-collapse of polynomial hierarchy [25, 26, 27].

The use of random IQP circuits for the verification protocol is problematic due to the anti-concentration property [28, 26]. To address this issue, the Shepherd-Bremner scheme employs an obfuscated quadratic-residue code (QRC) to construct the pair $(U_{\mathrm{IQP}}, \mathbf{s})$ [29]. While the Shepherd-Bremner scheme was experimentally attractive, it suffered from a drawback as its cryptographic assumptions were non-standard and lacked sufficient study compared to TCF-based protocols. This was especially apparent when in 2019 Kahanamoku-Meyer discovered a loophole in the Shepherd-Bremner scheme, enabling a classical prover to efficiently find the secret, which subsequently allows the prover to generate data to spoof the test [30]. Given the potential of IQP-based protocols to achieve verifiability beyond classical computing using fewer resources than, say, Shor's algorithm, it is crucial to investigate the possibility of extending and rectifying the Shepherd-Bremner construction.

In this work, we propose a new IQP-based protocol, which we refer to as the *stabilizer scheme*. Our construction allows the verifier to efficiently generate an IQP circuit, $U_{\mathrm{IQP}} = e^{i\pi H/8}$, and a secret, $\mathbf{s}$, so that the correlation function relative to the secret has a magnitude equal to $2^{-g/2}$, where g is a tunable integer. The stabilizer scheme is based on the interplay between IQP circuits, stabilizer formalism and coding theory, and it significantly strengthens previous constructions based on quadratic-residue codes [24] or random small IQP circuits [28]. Our characterization on IQP circuits builds upon and integrates several previous results [31, 32], which tackle this problem from the perspective of binary matroids and Tutte polynomials. In order to explore the classical security, we formulate the *Hidden Structured Code* problem, which captures the hardness of classical attacks based on secret extraction. Then, we investigate a general class of such classical attacks, which includes Kahanamoku-Meyer's attack as an instance. We give positive evidence that this class of classical attacks takes exponential time to generate the data with correct correlation relative to the secret. Specifically, we show that a generalization of Kahanamoku-Meyer's attack, named Linearity Attack, fails to break the stabilizer scheme if the parameters are chosen appropriately. Additionally, we have designed a new obfuscation technique called *column redundancy*, which can even be used to fix the recently-found weakness in the Shepherd-Bremner construction [33]. Specifically, Claim 3.1 in Ref. [30] states that the attack algorithm for the Shepherd-Bremner construction succeeds in $O(n^3)$ time on average, which turns out to be true only under certain parameter choices. This can be naturally rectified with proper parameter settings enabled by our column redundancy technique. Our results provide positive evidence for the security of the IQP-based verification protocols.

This paper is organized as follows. In the rest of the Introduction, we first give the general framework of IQP-based verification protocols. Then, we state our main results in more detail, followed by discussing the related works. In Section 2, we give the preliminaries, including stabilizer formalism, necessary results from coding theory and the Shepherd-Bremner construction. In Section 3, we give the characterization of the state generated by IQP circuits with $\theta = \pi/4$ and the correlation function $\langle \mathcal{Z}_{\mathbf{s}} \rangle$ with $\theta = \pi/8$. Then, in Section 4, we present the stabilizer construction for the IQP-based protocols. In Section 5, we analyze the classical security of the stabilizer scheme and explore the classical attacks based on secret extraction. Finally, we conclude and give open problems in Section 6.

1.1 IQP-based verification protocol

Here, we focus on a specific family of IQP circuits, the X program [24], where all local gates are diagonal in the Pauli-X basis. One can represent this family of IQP circuits by a time evolution of the Hamiltonian H, which consists of only products of Pauli X's. For example, for $H = X_1X_2X_4 + X_3X_4 + X_1X_3$, the corresponding IQP circuit is given by $U_{\mathrm{IQP}} = e^{i\theta H} = e^{i\theta X_1X_2X_4}e^{i\theta X_3X_4}e^{i\theta X_1X_3}$. In the general case, the evolution time for each term in H can be different, but we focus on the case where $\theta = \pi/8$ for all terms in this work. One can also use an m-by-n binary matrix to represent the IQP Hamiltonian, where m is the number of local terms and n is the number of qubits. Each row of the matrix represents one local term and the locations of 1's

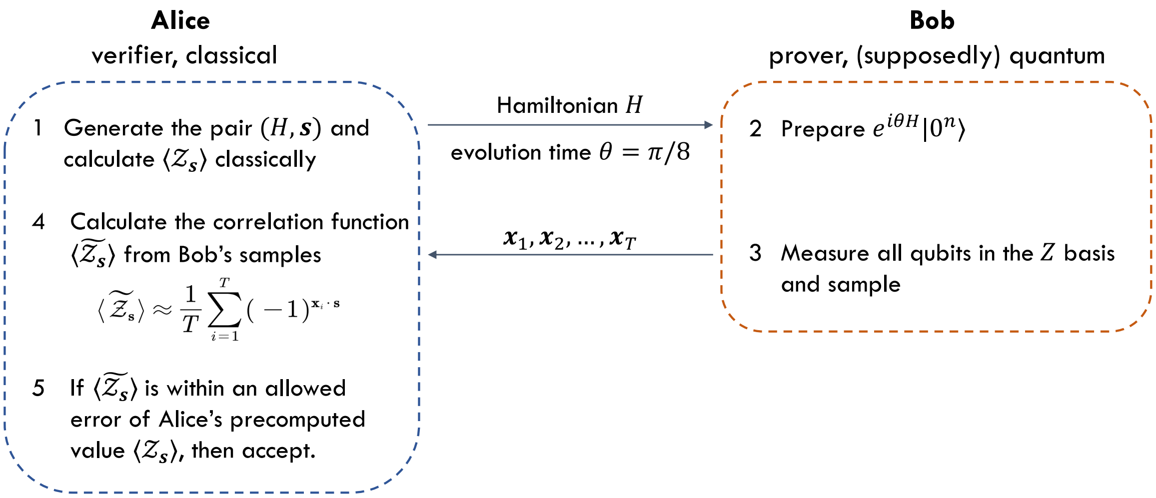

Figure 1: Schematic for IQP-based verification protocol in the case $\theta = \pi/8$.

indicate the qubits that it acts on. The matrix representation for H in the previous example is given by

$$\mathbf{H} = \begin{pmatrix} 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix} . \tag{1.1}$$

General framework. The general framework for the IQP-based verification protocol is shown in Fig. 1. Here, the verifier first generates the pair of IQP Hamiltonian H and the secret $\mathbf{s} \in \{0,1\}^n$. She computes the correlation function $\langle \mathcal{Z}_\mathbf{s} \rangle := \langle 0^n | U_{\text{IQP}}^\dagger \mathcal{Z}_\mathbf{s} U_{\text{IQP}} | 0^n \rangle$ relative to the secret, which can be achieved classically efficiently [28, 31]. Then, the classical description of the Hamiltonian H is sent to the prover, while the secret is kept on the verifier's side. The verifier also instructs the prover the evolution time for each term of the Hamiltonian. After that, the prover repeatedly prepares the state $e^{i\theta H}|0^n\rangle$, measures all qubits in the computational basis, and obtains a set of samples $\mathbf{x}_1, \ldots, \mathbf{x}_T \in \{0,1\}^n$, which will be sent back to the verifier. From the prover's measurement samples, the verifier estimates the correlation function relative to $\mathbf{s}$ by

$$\langle \widetilde{\mathcal{Z}_\mathbf{s}} \rangle := \frac{1}{T}\sum_{i=1}^{T}(-1)^{\mathbf{x}_i \cdot \mathbf{s}} . \tag{1.2}$$

If the value of $\langle \widetilde{\mathcal{Z}_\mathbf{s}} \rangle$ is within an allowed error of the ideal value $\langle \mathcal{Z}_\mathbf{s} \rangle$, then the verifier accepts the result and the prover passes the verification.

In order to ensure the effectiveness of the verification process, two key challenges must be addressed. The first one is to evaluate the ideal correlation function, so that the verifier can compare it with the value obtained from the prover's measurement outcomes. The second one is to design a suitable pair $(H, \mathbf{s})$, so that the correlation function $\langle \mathcal{Z}_\mathbf{s} \rangle$ is sufficiently away from zero. Otherwise, the verifier may need to request a super-polynomial number of samples from the prover to make the statistical error small enough, making the protocol inefficient.

Evaluating the correlation function. To evaluate the correlation function, first note that the Hamiltonian can be divided into two part $H = H_{\mathbf{s}} + R_{\mathbf{s}}$ based on the secret $\mathbf{s}$. Here, $H_{\mathbf{s}}$ anti-commutes with $\mathcal{Z}_{\mathbf{s}}$, i.e., $\{\mathcal{Z}_{\mathbf{s}}, H_{\mathbf{s}}\} = 0$, and the redundant part $R_{\mathbf{s}}$ commutes with $\mathcal{Z}_{\mathbf{s}}$, i.e., $[R_{\mathbf{s}}, \mathcal{Z}_{\mathbf{s}}] = 0$. Correspondingly, the matrix representations satisfy $\mathbf{H_s}\, \mathbf{s} = \mathbf{1}$ and $\mathbf{R_s}\, \mathbf{s} = \mathbf{0}$. Due to these commutation relations, the value of the correction function only depends on the $H_{\mathbf{s}}$, i.e. [28, 31],

$$\langle \mathcal{Z}_{\mathbf{s}} \rangle = \langle 0^n | e^{i2\theta H_{\mathbf{s}}} | 0^n \rangle \ . \tag{1.3}$$

Then, one can observe an intriguing point from this expression. When $\theta = \pi/8$, the IQP circuit is non-Clifford and there is complexity-theoretic evidence that the IQP circuits in this setting is hard to simulate classically [26]. However, $e^{i2\theta H_{\mathbf{s}}}$ becomes a Clifford circuit, which means that the correlation function can be computed classically efficiently! Indeed, $\langle \mathcal{Z}_{\mathbf{s}} \rangle = \langle 0^n | e^{i(\pi/4) H_{\mathbf{s}}} | 0^n \rangle$ actually corresponds to an amplitude of the Clifford circuit $e^{i(\pi/4)H_{\mathbf{s}}}$. In this way, the verifier can evaluate the correlation function efficiently using the Gottesman-Knill algorithm [34].

1.2 Main results

In this subsection, we briefly overview the main results of the paper in the following and refer the reader to later sections for the detailed analysis. The main objective of this work is to devise a new scheme of the IQP-based verification protocol that strengthens its classical security and invalidates the known attacks. To achieve this, we start by studying the properties of the state $e^{i\pi H/4} |0^n\rangle$. Given a binary matrix $\mathbf{H} = (\mathbf{c}_1, \ldots, \mathbf{c}_n)$, we first transform it into an IQP Hamiltonian H. By Theorem 3.1, the stabilizer tableau of $|\psi\rangle = e^{i\pi H/4} |0^n\rangle$ is given by $(\mathbf{G}, \mathbf{I}_n, \mathbf{r})$, where the X part is a Gram matrix $\mathbf{G} = \mathbf{H}^T\mathbf{H}$, the Z part is an identity matrix $\mathbf{I}_n$, and the phase column $\mathbf{r}$ depends on the Hamming weight of columns in $\mathbf{H}$.

Next, we compute the correlation function $|\langle \mathcal{Z}_{\mathbf{s}} \rangle|$ and connect it to a property of the code $\mathcal{C}_{\mathbf{s}}$ generated by columns of $\mathbf{H_s}$. Let $\mathcal{C}_{\mathbf{s}}^{\perp}$ be the dual code of $\mathcal{C}_{\mathbf{s}}$, $\mathcal{D}_{\mathbf{s}} := \mathcal{C}_{\mathbf{s}} \bigcap \mathcal{C}_{\mathbf{s}}^{\perp}$ be the self-dual intersection and consider $g := \dim(\mathcal{C}_{\mathbf{s}}) - \dim(\mathcal{D}_{\mathbf{s}})$. We then prove in Theorem 3.2 that the magnitude of the correlation function $|\langle \mathcal{Z}_{\mathbf{s}} \rangle|$ is $2^{-g/2}$ if the self-dual intersection $\mathcal{D}_{\mathbf{s}}$ is a doubly-even code, and 0 if it is an unbiased even code. Moreover, it can be proved that $\mathcal{D}_{\mathbf{s}}$ must be in one of the two cases, and thus the above gives a complete characterization of the magnitude of the correlation function. Interestingly, the g number happens to be the rank of the Gram matrix $\mathbf{G_s} = \mathbf{H_s}^T\mathbf{H_s}$ associated with $\mathbf{H_s}$ (Proposition 2.5), which also characterizes the overlap between $|0^n\rangle$ and $e^{i\pi H_{\mathbf{s}}/4} |0^n\rangle$ from a group-theoretic perspective (Proposition 2.1). Theorem 3.2 is an effective merging of a number of results that were first discussed by Shepherd in Ref. [31], with a particular focus on coding theory. Originally, Shepherd studied IQP circuits from the perspective of binary matroids, codes, and Tutte polynomials.

With these results established, the construction of $(\mathbf{H}, \mathbf{s})$ for the verification protocol can be formulated as follows. Let $\mathcal{H}_{n,m,g} = \{(\mathbf{H}, \mathbf{s})\}$ be a family of pairs of an IQP matrix $\mathbf{H} \in \mathbb{F}_2^{m \times n}$ and a secret $\mathbf{s} \in \mathbb{F}_2^n$ so that the corresponding correlation function satisfies $|\langle \mathcal{Z}_{\mathbf{s}} \rangle| = 2^{-g/2}$; the precise definition is presented in Definition 4.1. Here, the parameters n and m correspond to the size of the IQP circuits, and g corresponds to the value of the correlation function relative to the secret. Other than these three parameters, no other structure is imposed on the IQP circuits in this family. We give an efficient algorithm to sample random instances from $\mathcal{H}_{n,m,g}$, which we call the stabilizer construction (Meta-Algorithm 1).

Essentially, the stabilizer construction is to randomly generate an obfuscated code and a secret, so that the corresponding correlation function is sufficiently away from zero, to enable efficient verification. Specifically, we reduce this problem to sampling two matrices $\mathbf{D}$ and $\mathbf{F}$, so that $\mathbf{D}$ is a generator matrix of a random doubly-even code, and $\mathbf{F}$ consists of g random columns satisfying the constraints $\mathbf{D}^T\mathbf{F} = \mathbf{0}$ and $\mathrm{rank}(\mathbf{F}^T\mathbf{F}) = g$. Jointly, columns in $\mathbf{D}$ and $\mathbf{F}$ span a linear subspace that contains the all-ones vector, which must be a codeword because $\mathbf{H}_\mathbf{s}\,\mathbf{s} = \mathbf{1}$. We give an efficient algorithm to sample such matrices $\mathbf{D}$ and $\mathbf{F}$.

To explore the classical security, we consider a general class of classical attacks based on secret extraction. Given $(\mathbf{H}, \mathbf{s}) \in \mathcal{H}_{n,m,g}$, extracting the secret $\mathbf{s}$ from $\mathbf{H}$ leads to finding the hidden code $\mathcal{C}_\mathbf{s}$ from a larger obfuscated code. Such a hidden substructure problem seems hard for a classical computer, and we formulate the following conjecture.

Conjecture 1.1 (Hidden Structured Code (HSC) Problem)**.** *For certain appropriate choices of n, m, g, there exists an efficiently samplable distribution over instances $(\mathbf{H}, \mathbf{s})$ from the family $\mathcal{H}_{n,m,g}$, so that no polynomial-time classical algorithm can find the secret $\mathbf{s}$ given n, m and $\mathbf{H}$ as input, with high probability over the distribution on $\mathcal{H}_{n,m,g}$.*

To support this conjecture, we extend Kahanamoku-Meyer's attack to target general IQP circuits with $\theta = \pi/8$, and we call this attack the Linearity Attack. This generalized attack uses linear algebraic techniques to search for a candidate set of secrets, and performs classical sampling according to this candidate set. By choosing appropriate parameters, random instances drawn by our stabilizer scheme turns out to invalidate the Linearity Attack, since the search for the candidate set takes exponential time. As a result, the stabilizer scheme is secure against the Linearity Attack. Moreover, our analysis suggests that choosing a different set of parameters for the QRC-based construction can fix the recent loophole in the original Shepherd-Bremner scheme. This refutes the Claim 3.1 in Ref. [30], which states that the QRC-based construction can be efficiently broken classically in general.

1.3 Related works

The first explicit construction recipe of $(\mathbf{H}, \mathbf{s})$ for the case $\theta = \pi/8$ is given by Shepherd and Bremner [24]. In the their construction, $\mathbf{H}_\mathbf{s}$ is constructed from a specific error-correcting code, the quadratic-residue code (QRC) [29], which guarantees that the correlation function is always $1/\sqrt{2}$, a value sufficiently away from zero as desired. Formally, let $\mathcal{H}^{\mathrm{QRC}}_{n,m,q} = \{(\mathbf{H}, \mathbf{s})\}$ be a family of pairs of an IQP matrix $\mathbf{H} \in \mathbb{F}_2^{m\times n}$ and a secret $\mathbf{s}$ so that $\mathbf{H}_\mathbf{s}$ generates a QRC of length q (up to row permutations) and $\mathbf{H}$ is of full column rank. What the Shepherd-Bremner construction achieves is to randomly sample instances from $\mathcal{H}^{\mathrm{QRC}}_{n,m,q}$, where $n = (q+3)/2$.

However, it turns out that this set of parameters can only give easy instances. In Ref. [30], Kahanamoku-Meyer gave a secret-extraction attack (KM attack) against the Shepherd-Bremner construction. With his attack, a classical prover can find the secret $\mathbf{s}$ efficiently with high probability. Once the secret is found, the prover can easily pass the test by generating appropriately biased data in the direction of the secret, without the need of actually simulating the IQP circuits. In Ref. [28], Yung and Cheng proposed to circumvent the attack by starting with a small randomized IQP circuit and using the obfuscation technique in the Shepherd-Bremner scheme to hide that small IQP circuit [24]. The verifier cannot directly use a fully randomized IQP circuit because the correlation function will be close to zero for most choices of secrets in that case, due to the anti-concentration property of IQP circuits [26]. Small correlation functions make it difficult for the verifier to distinguish between an honest quantum prover and a cheating classical prover outputting random bit strings. This poses a challenge, to balance the security given by randomized constructions with the scale of the correlation functions that enables easy verification. This challenge is not fully resolved by the heuristic construction in Ref. [28].

In addition, Shepherd studied IQP circuits with tools of binary matroids and Tutte polynomials, and derived some related results to this work [31]. Specifically, the amplitude of the IQP circuit $\langle 0^n | e^{i\theta H} | 0^n \rangle$ is expressed in terms of the normalized Tutte polynomial, and its computational complexity is studied in various cases. When $\theta = \pi/4$, the magnitude of the related Tutte polynomial can be efficiently evaluated using Vertigan's algorithm [35], which is similar to the Gottesman-Knill algorithm [34]. This idea was further explored by Mann [32], which related computing the amplitude to the bicycle dimension and the Brown's invariant using results of Ref. [36]. But when $\theta = \pi/8$ (and any other values except for the multiple of $\pi/4$), computing the amplitude is $\#P$-hard in the worst case. Moreover, Ref. [31] also derived similar relation to Eq. (1.3), in the language of the normalized Tutte polynomial. Therefore, it was proved that the correlation function is efficiently classical computable when $\theta = \pi/8$, and suggests that this could be used to perform hypothesis test for access to quantum computers, although no new construction was proposed in Ref. [31].

2 Preliminaries

2.1 Notations

We mainly work on the field $\mathbb{F}_2$. We use bold capital letters such as $\mathbf{H}$ to denote a matrix and bold lower-case letters such as $\mathbf{s}$ to denote a vector. If not stated otherwise, a vector is referred to as a column vector, and a row vector will be added the transpose symbol, like $\mathbf{p}^T$. The (Hamming) weight of a vector $\mathbf{x}$ is denoted as $|\mathbf{x}|$. The inner product between two vectors $\mathbf{x}$ and $\mathbf{s}$ is denoted as $\mathbf{x} \cdot \mathbf{s}$; sometimes we will also use $\mathbf{H} \cdot \mathbf{s}$ to denote the matrix multiplication. We use $\mathrm{col}(\mathbf{H})$ and $\mathrm{row}(\mathbf{H})$ to denote the columns and rows of a matrix $\mathbf{H}$, respectively. We use $c(\mathbf{H})$ and $r(\mathbf{H})$ to denote the number of columns and the number of rows of a matrix $\mathbf{H}$, respectively. The rank of a matrix $\mathbf{H}$ is denoted as $\mathrm{rank}(\mathbf{H})$. We use $\ker(\mathbf{H})$ to denote the kernel space of $\mathbf{H}$, i.e., the space of vector $\mathbf{v}$ such that $\mathbf{Hv} = \mathbf{0}$. We call two square matrices $\mathbf{A}$ and $\mathbf{B}$ congruent if there exists an invertible matrix $\mathbf{Q}$ satisfying $\mathbf{A} = \mathbf{Q}^T\mathbf{BQ}$, denoted as $\mathbf{A} \sim_c \mathbf{B}$. We call such an transformation *congruent transformation*.

The all-ones vector will be denoted as $\mathbf{1}$, with its dimension inspected from the context; the similar rule applies to the all-zeros vector (or matrix) $\mathbf{0}$. The $n \times n$ identity matrix is denoted as $\mathbf{I}_n$. For a vector $\mathbf{x}$, we define its support as $\{j : x_j = 1\}$. We define $[n] := \{1, 2, \ldots, n\}$. If not stated otherwise, a full-rank matrix is referred to a matrix with full column rank.

We denote the linear subspace spanned by a set of vectors $\{\mathbf{c}_1, \ldots, \mathbf{c}_k\}$ as $\langle \mathbf{c}_1, \ldots, \mathbf{c}_k \rangle$. Given linear subspaces $V = \langle \mathbf{c}_1, \ldots, \mathbf{c}_l \rangle$ and $U = \langle \mathbf{c}_1, \ldots, \mathbf{c}_k \rangle$ with $k < l$, we denote the complement subspace of U in V with respect to the basis $\{\mathbf{c}, \ldots, \mathbf{c}_l\}$ by $(V/U)_{\mathbf{c}_1, \ldots, \mathbf{c}_l}$; namely, $(V/U)_{\mathbf{c}_1, \ldots, \mathbf{c}_l} := \langle \mathbf{c}_{k+1}, \ldots, \mathbf{c}_l \rangle$. Usually, we are not interested in a specific basis, so we use V/U to denote a random complement subspace of U in V, i.e., $V/U \leftarrow_{\mathcal{R}} \{\langle \mathbf{c}_{k+1}, \ldots, \mathbf{c}_l \rangle : V = \langle \mathbf{c}_1, \ldots, \mathbf{c}_l \rangle, U = \langle \mathbf{c}_1, \ldots, \mathbf{c}_k \rangle\}$, where $\leftarrow_{\mathcal{R}}$ denotes a random instance from a set. We let $V \backslash U := \{\mathbf{v} : \mathbf{v} \in V, \mathbf{v} \notin U\}$ be the ordinary complement of two sets.

2.2 Stabilizer formalism

Overlap of two stabilizer states. Given two stabilizer states $|\psi\rangle$ and $|\phi\rangle$, let $\mathrm{Stab}(|\psi\rangle)$ and $\mathrm{Stab}(|\phi\rangle)$ be their stabilizer groups, respectively, which are subgroups of the n-qubit Pauli group. Let $\{P_1, \dots, P_n\}$ be the generators of $\mathrm{Stab}(|\psi\rangle)$ and $\{Q_1, \dots, Q_n\}$ be those of $\mathrm{Stab}(|\phi\rangle)$. Note that the set of generators is not unique. Then, the overlap $|\langle\psi|\phi\rangle|$ is determined by their stabilizer groups [37].

Proposition 2.1 ([37]). *Let $|\psi\rangle$ and $|\phi\rangle$ be two stabilizer states. Then, $\langle\psi|\phi\rangle = 0$ if their stabilizer groups contain the same Pauli operator of the opposite sign. Otherwise, $|\langle\psi|\phi\rangle| = 2^{-g/2}$, where g is the minimum number of different generators over all possible choices.*

For completeness, we provide an alternative proof in Appendix A. In particular, this implies that $\langle \mathcal{Z}_{\mathbf{s}} \rangle = \langle 0^n | e^{i\pi H_{\mathbf{s}}/4} | 0^n \rangle$ has magnitude either 0 or $2^{-g/2}$, where $n - g$ is the maximum number of independent Pauli-Z products in the stabilizer group of $e^{i\pi H_{\mathbf{s}}/4} |0^n\rangle$.

Tableau representation. A stabilizer state or circuit can be represented by a stabilizer tableau, which is an n-by-$(2n+1)$ binary matrix. The idea is to use $2n+1$ bits to represent each generator of the stabilizer group. First, a single-qubit Pauli operator can be represented by (x, z); $(0, 0)$ corresponds to I, $(1, 0)$ corresponds to X, $(0, 1)$ corresponds to Z and $(1, 1)$ corresponds to Y. For stabilizer generators, the phase can only be ± 1 since the stabilizer group does not contain $-I$. So, one can use an extra bit r to represent the phase; $r = 0$ is for $+1$ while $r = 1$ is for -1. Then, an n-qubit stabilizer generator can be represented by $2n + 1$ bits,

$$(x_1, \dots, x_n, z_1, \dots, z_n, r) \; . \tag{2.1}$$

For example, the vector for $-X_1 Z_2$ is $(1, 0, 0, 1, 1)$. Any stabilizer state can be specified by n stabilizer generators, which commute with each other. Therefore, the state is associated with the following tableau,

$$\begin{pmatrix} x_{11} & \cdots & x_{1n} & z_{11} & \cdots & z_{1n} & r_1 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{n1} & \cdots & x_{nn} & z_{n1} & \cdots & z_{nn} & r_n \end{pmatrix} , \tag{2.2}$$

whose rows define the stabilizer generators. The first n columns are called the X part, the $(n+1)$-th to $2n$-th columns are called the Z part, and the last column are called the phase column of the stabilizer tableau. As an example, the $|0^n\rangle$ state is stabilized by $\langle Z_1, \dots, Z_n \rangle$, and its stabilizer tableau is given by,

$$\begin{pmatrix} 0 & \cdots & 0 & 1 & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 0 & 0 & \cdots & 1 & 0 \end{pmatrix} . \tag{2.3}$$

We will call it the standard stabilizer tableau of $|0^n\rangle$.

2.3 Coding theory

We present some results regarding coding theory here, with the proof presented in Appendix B. We only consider linear codes over $\mathbb{F}_2$ in this paper. A linear code, or simply a code $\mathcal{C}$ of length m

is a linear subspace of $\mathbb{F}_2^m$. One can use a generator matrix $\mathbf{H}$ to represent a code, with its columns spanning the codespace $\mathcal{C}$. The dual code is defined as $\mathcal{C}^\perp := \{\mathbf{v} \in \mathbb{F}_2^m : \mathbf{v} \cdot \mathbf{w} = 0 \text{ for } \mathbf{w} \in \mathcal{C}\}$. The dual code of a linear code is also a linear code. It is not hard to see that $\mathcal{C}^\perp = \ker(\mathbf{H}^T)$, which implies $\dim(\mathcal{C}) + \dim(\mathcal{C}^\perp) = m$. A code $\mathcal{C}$ is weakly self-dual if $\mathcal{C} \subseteq \mathcal{C}^\perp$ and (strictly) self-dual if $\mathcal{C} = \mathcal{C}^\perp$, in which case $\dim(\mathcal{C}) = m/2$.

A code $\mathcal{C}$ is an even code if all codewords have even Hamming weight and a doubly-even code if all codewords have Hamming weight a multiple of 4. It is not hard to show that a doubly-even code is a weakly self-dual code. Moreover, we have the following proposition.

Proposition 2.2. *The all-ones vector is a codeword of $\mathcal{C}$ if and only if its dual code $\mathcal{C}^\perp$ is an even code.*

We define the notion of (un)biased even codes, which will be useful in the stabilizer characterization of IQP circuits (Section 3).

Definition 2.3. A code $\mathcal{C}$ is called a *biased even code* if it is an even code where the number of codewords with Hamming weight 0 and 2 modulo 4 are not equal. It is called an *unbiased even code* otherwise.

Let the (maximum) self-dual subspace of $\mathcal{C}$ be $\mathcal{D} := \mathcal{C} \bigcap \mathcal{C}^\perp$, which is itself a weakly self-dual code. Note that $\mathcal{D}$ must be an even code, since all codewords are orthogonal to themselves and hence have even Hamming weight. We have the following lemma.

Lemma 2.4. *A weakly self-dual even code is either a doubly-even code or an unbiased even code. For the former case, all columns of its generator matrix have weight 0 modulo 4 and are orthogonal to each other. For the latter case, there is at least one column in the generator matrix with weight 2 modulo 4.*

One can apply a basis change to the generator matrix $\mathbf{H}$, resulting in $\mathbf{HQ}$, where $\mathbf{Q}$ is an invertible matrix. This will not change the code $\mathcal{C}$. Define the Gram matrix of the generator matrix by $\mathbf{G} := \mathbf{H}^T\mathbf{H}$. A basis change on $\mathbf{H}$ transforms $\mathbf{G}$ into $\mathbf{Q}^T\mathbf{G}\mathbf{Q}$, which is a congruent transformation. The rank of Gram matrix is also an invariant under basis change. It can be related to the code $\mathcal{C}$ in the following way.

Proposition 2.5. *Given a generator matrix $\mathbf{H}$, let its Gram matrix be $\mathbf{G} = \mathbf{H}^T\mathbf{H}$ and the generated code be $\mathcal{C}$. Let $\mathcal{D} = \mathcal{C} \bigcap \mathcal{C}^\perp$, where $\mathcal{C}^\perp$ is the dual code of $\mathcal{C}$. Then, $\operatorname{rank}(\mathbf{G}) = \dim(\mathcal{C}) - \dim(\mathcal{D})$.*

2.4 Shepherd-Bremner construction

In the Shepherd-Bremner construction, the part $\mathbf{H_s}$ is constructed from the quadratic-residue code. The quadratic residue code is a cyclic code. Its cyclic generator has 1 in the j-th position if j is a non-zero quadratic residue modulo q. The size parameter q of the QRC is a prime number and $q + 1$ is required to be a multiple of eight [24]. For $q = 7$, the cyclic generator reads $(1, 1, 0, 1, 0, 0, 0)^T$, because $j = 1, 2, 4$ are quadratic residues modulo 7. The basis for the codespace of QRC is generated by rotating the cyclic generator, which is the last 4 columns of the following matrix,

$$\mathbf{H}_\mathbf{s}^{\mathrm{QRC}} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{pmatrix} . \tag{2.4}$$

The first column is added so that the secret is easy to find, i.e., $\mathbf{s} = (1, 0, 0, 0, 0)^T$.

After obtaining the initial $\mathbf{H}_\mathbf{s}^{\mathrm{QRC}}$, the verifier needs to hide the secret and make the IQP circuit look random, while leaving the value of the correlation function unchanged. In the Shepherd-Bremner construction, the verifier will first add redundant rows $\mathbf{R}_\mathbf{s}$, which are rows that are orthogonal to $\mathbf{s}$, to obtain the full IQP matrix

$$\mathbf{H} = \begin{pmatrix} \mathbf{H}_\mathbf{s}^{\mathrm{QRC}} \\ \mathbf{R}_\mathbf{s} \end{pmatrix} . \tag{2.5}$$

Its corresponding Hamiltonian $R_\mathbf{s}$ commutes with $\mathcal{Z}_\mathbf{s}$ and hence will not affect the correlation function. After initializing $\mathbf{H}$ and $\mathbf{s}$, the verifier needs to apply obfuscation to hide the secret. The obfuscation is achieved by randomly permuting rows in $\mathbf{H}$ and performing column operations to $\mathbf{H}$ and changing $\mathbf{s}$ accordingly.

Definition 2.6 (Obfuscation)**.** Given an instance $(\mathbf{H}, \mathbf{s})$, the obfuscation is defined as the transformation

$$\mathbf{H} \leftarrow \mathbf{PHQ} \qquad\qquad \mathbf{s} \leftarrow \mathbf{Q}^{-1}\mathbf{s} \,, \tag{2.6}$$

where $\mathbf{P}$ is a random row-permutation matrix and $\mathbf{Q}$ is a random invertible matrix.

Note that row permutations will not change the value of the correlation function, since the gates in IQP circuits commute with each other. As for the column operations, it can be shown that if the secret $\mathbf{s}$ is transformed accordingly, to maintain the inner-product relation with the rows in $\mathbf{H}$, then the value of the correlation function remains unchanged [24, 28].

In the Shepherd-Bremner scheme [24], the measure of success is given by the probability bias $\mathcal{P}_{\mathbf{s}\perp} := \sum_{\mathbf{x}\cdot\mathbf{s}=0} p(\mathbf{x})$, the probability of receiving bit strings that are orthogonal to the secret $\mathbf{s}$, where $p(\mathbf{x})$ is the output probability of the IQP circuit. This measure is equivalent to the correlation function, since $\mathcal{P}_{\mathbf{s}\perp} = \frac{1}{2}(\langle \mathcal{Z}_\mathbf{s} \rangle + 1)$ [31, 38]. Due to the properties of QRC, $\langle \mathcal{Z}_\mathbf{s} \rangle$ always equals $1/\sqrt{2}$ (in terms of probability bias, 0.854).

3 Stabilizer characterization of IQP circuits

In this section, we establish the connection between IQP circuits, stabilizer formalism and coding theory, which turns out to be useful in constructing the IQP circuits for the verification protocol. For $\theta = \pi/8$, we show that the stabilizer tableau of the Clifford operation $e^{i2\theta H_\mathbf{s}}$ has a nice structure that allows us to determine the value of $\langle \mathcal{Z}_\mathbf{s} \rangle = \langle 0^n | e^{i2\theta H_\mathbf{s}} | 0^n \rangle$ efficiently. As an application, we analyze the Shepherd-Bremner construction with this framework.

We first give the form of the stabilizer tableau of $e^{i\pi H/4} |0^n\rangle$.

Theorem 3.1. *Given a binary matrix $\mathbf{H} = (\mathbf{c}_1, \ldots, \mathbf{c}_n)$ and transforming it into an IQP Hamiltonian H, the stabilizer tableau of the state $|\psi\rangle = e^{i\pi H/4} |0^n\rangle$ can be expressed as,*

$$\left(\begin{array}{ccc|ccc|c} \mathbf{c}_1 \cdot \mathbf{c}_1 & \cdots & \mathbf{c}_1 \cdot \mathbf{c}_n & 1 & \cdots & 0 & r_1 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{c}_n \cdot \mathbf{c}_1 & \cdots & \mathbf{c}_n \cdot \mathbf{c}_n & 0 & \cdots & 1 & r_n \end{array}\right). \tag{3.1}$$

Here, if one uses $00, 01, 10, 11$ *to represent* $|\mathbf{c}_j| = 0, 1, 2, 3 \pmod 4$*, then* r_j *is equal to the first bit.*

This theorem can be proved by starting from the standard tableau of $|0^n\rangle$, and keeping track of the stabilizer tableau after applying each terms of $e^{i\pi H/4}$ (i.e., each row of $\mathbf{H}$). The complete proof is delayed to Appendix C. We will call Eq. (3.1) the IQP (stabilizer) tableau and it is of the form $(\mathbf{G}, \mathbf{I}_n, \mathbf{r})$. We apply the above theorem to $\mathbf{H}_\mathbf{s}$, in which case the X part is $\mathbf{G}_\mathbf{s} = \mathbf{H}_\mathbf{s}^T \mathbf{H}_\mathbf{s}$.

Next, we relate the correlation function to the code generated by $\mathbf{H}_\mathbf{s}$, denoted as $\mathcal{C}_\mathbf{s}$. Note that $\mathbf{H}_\mathbf{s}\mathbf{s} = \mathbf{1}$ means that the all-ones vector is a codeword of $\mathcal{C}_\mathbf{s}$. From Proposition 2.2, this means that the dual code $\mathcal{C}_\mathbf{s}^\perp$ is an even code and the intersection $\mathcal{D}_\mathbf{s} := \mathcal{C}_\mathbf{s} \bigcap \mathcal{C}_\mathbf{s}^\perp$ is a weakly self-dual even code. Then, $\mathcal{D}_\mathbf{s}$ will be either a doubly-even code or an unbiased even code, according to Lemma 2.4.

Theorem 3.2. *Given an IQP matrix $\mathbf{H}_\mathbf{s}$ and a vector $\mathbf{s}$, so that $\mathbf{H}_\mathbf{s}\,\mathbf{s} = \mathbf{1}$. Denote the code generated by columns of $\mathbf{H}_\mathbf{s}$ by $\mathcal{C}_\mathbf{s}$ and its dual code by $\mathcal{C}_\mathbf{s}^\perp$. Let $\mathcal{D}_\mathbf{s} := \mathcal{C}_\mathbf{s} \bigcap \mathcal{C}_\mathbf{s}^\perp$. Then, transforming $\mathbf{H}_\mathbf{s}$ into an IQP Hamiltonian $H_\mathbf{s}$, the magnitude of the correlation function $\langle \mathcal{Z}_\mathbf{s} \rangle = \langle 0^n | e^{i\pi H_\mathbf{s}/4} | 0^n \rangle$ is $2^{-g/2}$ if $\mathcal{D}_\mathbf{s}$ is a doubly-even code and zero if $\mathcal{D}_\mathbf{s}$ is an unbiased even code. Here, $g := \dim(\mathcal{C}_\mathbf{s}) - \dim(\mathcal{D}_\mathbf{s})$ is also the rank of the Gram matrix $\mathbf{G}_\mathbf{s} = \mathbf{H}_\mathbf{s}^T \mathbf{H}_\mathbf{s}$.*

We leave the proof in Appendix C. Interestingly, from a group-theoretic perspective, the rank of the Gram matrix g is also the minimum number of different generators over all possible choices of the stabilizer groups between $|0^n\rangle$ and $e^{i\pi H_\mathbf{s}/4} |0^n\rangle$ (Proposition 2.1). Furthermore, we note that this result integrates several results in Ref. [31] concisely, with a particular focus on coding theory, so that it aligns better with our objective of constructing IQP circuits for the verification protocol. Ref. [31] studies the IQP circuits with $\theta = \pi/4$ with a reworking of Vertigan's algorithm for evaluating the magnitude of the Tutte polynomial of a binary matroid at the point $(-i, i)$ [35]. There, the amplitude $\langle \mathbf{x} | e^{i\theta H} | 0^n \rangle$ is considered for $\theta = \pi/4$ and any IQP Hamiltonian H, where the all-ones vector may not be a codeword of the code generated by the binary matrix $\mathbf{H}$. Such an amplitude has been further studied in Ref. [32], which gives the expression of the phase of the amplitude by applying results of Ref. [36]. In the language of binary matroids, the dual intersection $\mathcal{D}_\mathbf{s}$ is the bicycle space of the matroid represented by $\mathbf{H}_\mathbf{s}$ and its dimension $\dim(\mathcal{D}_\mathbf{s})$ is also known as the bicycle dimension [35, 32]. Finally, we note that although computing the magnitude suffices for our later construction, the sign of the correlation function can also be computed efficiently, as shown in Ref. [32]. In addition, when $g = O(\log n)$, the correlation function has an inverse polynomial scaling. In this case, one can use the random sampling algorithm in Ref. [28] to determine the sign efficiently.

To show the usefulness of the stabilizer characterization, we apply these two theorems to analyze the Shepherd-Bremner construction. Combined with the properties of QRC, we have the following corollary (with proof presented in Appendix C).

Corollary 3.3. *Let q be a prime such that 8 divides $q + 1$. Let $\mathbf{H}_\mathbf{s}^{\mathrm{QRC}}$ be a matrix whose first column is $\mathbf{1}$ (of length q), and whose remaining columns are the basis of the quadratic-residue code of length q, formed* *by the cyclic generator (i.e., in the form of Eq. (2.4)). Then, translating $\mathbf{H}_\mathbf{s}^{\mathrm{QRC}}$ into an IQP Hamiltonian $H_\mathbf{s}$, the stabilizer tableau of $|\psi_\mathbf{s}\rangle = e^{i\pi H_\mathbf{s}/4}|0^n\rangle$ can be expressed as the following form,*

$$\left(\begin{array}{ccc|ccc|c} 1 & \cdots & 1 & 1 & \cdots & 0 & 1 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & \cdots & 1 & 1 & \cdots & 1 & 1 \end{array}\right) . \tag{3.2}$$

As a result, the corresponding stabilizer group is given by,

$$\langle -Y_1 X_2 \cdots X_n, -X_1 Y_2 X_3 \cdots X_n, \ldots, -X_1 X_2 \cdots X_{n-1} Y_n \rangle , \tag{3.3}$$

where $n = (q+3)/2$. Moreover, the correlation function $\langle \mathcal{Z}_\mathbf{s} \rangle = \langle 0^n | \psi_\mathbf{s} \rangle$ has a magnitude $1/\sqrt{2}$.

4 Stabilizer construction

In this section, we present the stabilizer construction, which is a systematic way to construct IQP circuits with $\theta = \pi/8$ for verification. In fact, the goal is to generate a pair $(\mathbf{H}, \mathbf{s})$, such that they satisfy certain conditions, which stem from Theorem 3.2. We first define the family of pairs that we would like to sample from.

Definition 4.1. Let $\mathcal{H}_{n,m,g} = \{(\mathbf{H}, \mathbf{s})\}$ be a family of pairs of an IQP matrix $\mathbf{H} \in \mathbb{F}_2^{m \times n}$ and a secret $\mathbf{s} \in \mathbb{F}_2^n$ satisfying the following conditions. (1) $\mathcal{D}_\mathbf{s} = \mathcal{C}_\mathbf{s} \bigcap \mathcal{C}_\mathbf{s}^\perp$ is a doubly-even code, where $\mathcal{C}_\mathbf{s}$ is the code generated by columns of $\mathbf{H}_\mathbf{s}$ and $\mathcal{C}_\mathbf{s}^\perp$ is its dual code; (2) $\mathrm{rank}(\mathbf{H}_\mathbf{s}^T \mathbf{H}_\mathbf{s}) = g$; (3) $\mathrm{rank}(\mathbf{H}) = n$.

In this definition, the size of the IQP circuits are determined by n and m, which correspond to the number of qubits and gates, respectively. Additionally, condition (1) is to guarantee that the correlation function $\langle \mathcal{Z}_\mathbf{s} \rangle$ corresponding to instances of $\mathcal{H}_{n,m,g}$ is nonzero, and condition (2) states that its magnitude is given by $2^{-g/2}$. Therefore, the family $\mathcal{H}_{n,m,g}$ includes all instances of IQP circuits of a certain size that have correlation function $\pm 2^{-g/2}$ with respect to some secret $\mathbf{s}$. Note that the rank of the Gram matrix $\mathbf{H}_\mathbf{s}^T \mathbf{H}_\mathbf{s}$ should be $g = O(\log n)$ for the protocol to be practical. The reason for considering IQP matrices $\mathbf{H}$ with full column rank will be made clear when we discuss the classical security of the IQP-based verification protocol (Section 5.1.2).

Moreover, we give an efficient classical sampling algorithm to sample instances from $\mathcal{H}_{n,m,g}$, which is the stabilizer construction (Meta-Algorithm 1).

Theorem 4.2. *There exists an efficient classical sampling algorithm that sample from $\mathcal{H}_{n,m,g}$, given the parameters n, m and g.*

For the algorithmic purpose, we set two additional parameters, m_1 and d, which are the number of rows in $\mathbf{H}_\mathbf{s}$ and the dimension of $\mathcal{D}_\mathbf{s}$, respectively. These are random integers satisfying certain natural constraints (see Appendix D). The rank of $\mathbf{H}_\mathbf{s}$ is then equal to $r = g + d$. The stabilizer construction works by sampling $\mathbf{H}_\mathbf{s}$ and $\mathbf{R}_\mathbf{s}$ in certain 'standard forms', up to row permutations and column operations. Note that the 'standard forms' of $\mathbf{H}_\mathbf{s}$ and $\mathbf{R}_\mathbf{s}$ are not necessarily unique.

We first discuss $\mathbf{R}_\mathbf{s}$. To ensure that $\mathrm{rank}(\mathbf{H}) = n$, observe that in any $\mathbf{H}$ of full column rank, the redundant rows $\mathbf{R}_\mathbf{s}$ can always be transformed by row permutations into a form, where

Parameters: n, m, g
Output: $(\mathbf{H}, \mathbf{s}) \in \mathcal{H}_{n,m,g}$

1: Randomly sample m_1 and d with certain constraints ▷ Appendix D
2: Sample $\mathbf{D} \in \mathbb{F}_2^{m_1 \times d}$ and $\mathbf{F} \in \mathbb{F}_2^{m_1 \times g}$ satisfying certain conditions ▷ Appendix D
3: Initialize $\mathbf{H_s} \leftarrow (\mathbf{F}, \mathbf{D}, \mathbf{0}_{m_1 \times (n-r)})$, where $r = g + d$
4: Sample a secret $\mathbf{s}$ from the solutions of $\mathbf{H_s}\,\mathbf{s} = \mathbf{1}$
5: $\mathbf{H} \leftarrow \begin{pmatrix} \mathbf{H_s} \\ \mathbf{R_s} \end{pmatrix}$, where $\mathbf{R_s}$ is a random matrix with $m - m_1$ rows satisfying $\mathbf{R_s s} = \mathbf{0}$ and $\operatorname{rank}(\mathbf{H}) = n$
6: Perform obfuscation as in Definition 2.6

Meta-Algorithm 1: Stabilizer construction

the first $n - r$ rows form a basis of $\mathbb{F}_2^n$ together with the rows in $\mathbf{H_s}$. Therefore, up to row permutations, the first $n - r$ rows of $\mathbf{R_s}$ are sampled to be random independent rows that are orthogonal to $\mathbf{s}$ and lie outside the row space of $\mathbf{H_s}$. The remaining rows in $\mathbf{R_s}$ are random rows orthogonal to $\mathbf{s}$.

Next, we discuss sampling $(\mathbf{H_s}, \mathbf{s})$, which is the core of the stabilizer construction. Essentially, we want to randomly generate a (possibly redundant) generator matrix $\mathbf{H_s}$ of a code $\mathcal{C}_\mathbf{s}$, so that its dimension is r, its intersection $\mathcal{D}_\mathbf{s}$ with the dual code is a doubly-even code with dimension $d = r - g$ and the all-ones vector is a codeword. The last condition guarantees that a secret $\mathbf{s}$ can always be found. Note that, we allow $\operatorname{rank}(\mathbf{H_s}) < n$. That is, we allow $\mathbf{H_s}$ to be a "redundant" generator matrix of $\mathcal{C}_\mathbf{s}$, instead of a full-rank one. This is called adding column redundancy to the full-rank generator matrix of $\mathcal{C}_\mathbf{s}$, because after the obfuscation process, there will be redundant linear combinations in the columns of $\mathbf{H_s}$. We give a more formal discussion of column redundancy in Appendix E.

For such a generator matrix $\mathbf{H_s}$, there is an invertible matrix $\mathbf{Q}$ to perform a basis change so that

$$\mathbf{H_s Q} = (\mathbf{F}, \mathbf{D}, \mathbf{0}_{m_1 \times (n-r)}) \,, \tag{4.1}$$

where $\mathbf{D} \in \mathbb{F}_2^{m_1 \times d}$ is a generator matrix of the doubly-even code $\mathcal{D}_\mathbf{s}$, and columns in $\mathbf{F} \in \mathbb{F}_2^{m_1 \times g}$ span $\mathcal{C}_\mathbf{s}/\mathcal{D}_\mathbf{s}$. In addition, it can be shown that $\operatorname{rank}(\mathbf{F}^T\mathbf{F}) = \operatorname{rank}(\mathbf{Q}^T\mathbf{H}_\mathbf{s}^T\mathbf{H_s Q}) = \operatorname{rank}(\mathbf{H}_\mathbf{s}^T\mathbf{H_s}) = g$. Moreover, although there might be no unique standard form of $\mathbf{H_s}$, the Gram matrix has a unique standard form. First note that row permutations have no effect on the Gram matrix, since $\mathbf{P}^T\mathbf{P} = \mathbf{I}$ for a permutation matrix $\mathbf{P}$. So we focus on column operations. As shown in Ref. [39], there exists an invertible matrix $\mathbf{Q}$, so that $\mathbf{Q}^T\mathbf{H}_\mathbf{s}^T\mathbf{H_s Q} = \operatorname{diag}(\mathbf{I}_g, \mathbf{0})$ or $\operatorname{diag}\left(\bigoplus_{i=1}^{g/2} \mathbf{J}, \mathbf{0}\right)$, depending on whether at least one diagonal element of $\mathbf{H}_\mathbf{s}^T\mathbf{H_s}$ is 1 or not, where $\mathbf{J} := \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$. However, for the construction purpose, we need to ensure that the all-ones vector is a codeword of $\mathcal{C}_\mathbf{s}$. Therefore, in Appendix D, we give a slightly different standard form of $\mathbf{H}_\mathbf{s}^T\mathbf{H_s}$, which can be achieved by $\mathbf{H_s}$ in the form of $(\mathbf{F}, \mathbf{D}, \mathbf{0})$.

In summary, sampling $(\mathbf{H_s}, \mathbf{s})$ is reduced to generating an $\mathbf{H_s} = (\mathbf{F}, \mathbf{D}, \mathbf{0})$ so that the Gram matrix $\mathbf{H}_\mathbf{s}^T\mathbf{H_s}$ is in the standard form presented in Appendix D. Then, a secret $\mathbf{s}$ is sampled from the solutions of $\mathbf{H_s}\,\mathbf{s} = \mathbf{1}$. Sampling such an $\mathbf{H_s}$ is further reduced to sampling $\mathbf{D}$ and $\mathbf{F}$, so that $\mathbf{D}$ is a generator matrix for a random doubly-even code and $\mathbf{F}$ is a random matrix satisfying $\mathbf{D}^T\mathbf{F} = \mathbf{0}$, $\text{rank}(\mathbf{F}^T\mathbf{F}) = g$ and that $\mathbf{1}$ is in the column space of $(\mathbf{F}, \mathbf{D})$. We claim that sampling such $\mathbf{D}$ and $\mathbf{F}$ can be done efficiently, with details deferred to Appendix D.

5 Classical attacks and security

In this section, we examine the classical security of our protocol, i.e., the possibility that an efficient classical prover can pass the test. A straightforward classical attack is to simulate the IQP circuit sent by the verifier. We do not expect this to be efficient, since there is generally no structure to be exploited by a classical simulation algorithm. For example, due to the obfuscation as in Eq. (2.6), the geometry of the IQP circuit can be arbitrary, which implies that the treewidth in a tensor network algorithm cannot be easily reduced [40].

Here, we focus on another class of classical attacks based on extracting secrets. Given an IQP matrix $\mathbf{H}$, once the hidden secret $\mathbf{s}$ is found, a classical prover can first calculate the correlation function $\langle \mathcal{Z}_\mathbf{s} \rangle$ efficiently. Then, he generates a sample $\mathbf{x}$ which is orthogonal to $\mathbf{s}$ with probability $(1 + \langle \mathcal{Z}_\mathbf{s} \rangle)/2$ and not orthogonal to $\mathbf{s}$ with probability $(1 - \langle \mathcal{Z}_\mathbf{s} \rangle)/2$. The generated samples will have the correct correlation with the secret $\mathbf{s}$ and hence pass the test. Kahanamoku-Meyer's attack algorithm for the Shepherd-Bremner construction is an instance of this class [30].

But generally, this attack may not be efficient. From a code perspective, the stabilizer construction is to sample a random code satisfying certain constraints, and hide it by adding redundancy and performing obfuscation. Finding the secret allows one to find the hidden subcode, which should be a hard problem in general. In particular, we formulate the following conjecture.

Conjecture 5.1 (Hidden Structured Code (HSC) Problem, Restatement of Conjecture 1.1)**.** *For certain appropriate choices of n, m, g, there exists an efficiently samplable distribution over instances $(\mathbf{H}, \mathbf{s})$ from the family $\mathcal{H}_{n,m,g}$, so that no polynomial-time classical algorithm can find the secret $\mathbf{s}$ given n, m and $\mathbf{H}$ as input, with high probability over the distribution on $\mathcal{H}_{n,m,g}$.*

Naturally, sampling instances with uniform distribution from $\mathcal{H}_{n,m,g}$ is more favorable, since it does not put any bias on specific instances. For the underlying distribution induced by the stabilizer construction (Meta-Algorithm 1), it seems that it is uniform or close to uniform, as the output instances are random instances satisfying certain natural constraints imposed by the structure of the family $\mathcal{H}_{n,m,g}$. Though, we do not have a rigorous proof for this claim. Moreover, a similar conjecture was given in Ref. [24] for the family $\mathcal{H}^{\text{QRC}}_{n,m,q}$, where the problem is to decide whether a given $\mathbf{H}$ is from the family $\mathcal{H}^{\text{QRC}}_{n,m,q}$ or not. They conjectured that such a problem is NP-complete. Here, to better align with the classical attack, we consider the problem of finding the secret $\mathbf{s}$ instead.

To support Conjecture 5.1, we first generalize Kahanamoku-Meyer's attack algorithm to target any IQP-based verification protocols with $\theta = \pi/8$. We show that this generalized attack, named the Linearity Attack, fails to break our construction. Furthermore, our analysis reveals that the loophole of the original Shepherd-Bremner construction stems from an improper choice of parameters. The Shepherd-Bremner construction can be improved by the column redundancy technique, which enables random sampling from the family $\mathcal{H}^{\text{QRC}}_{n,m,q}$ with any possible parameters and thereby fixes the loophole.

procedure EXTRACTSECRET($\mathbf{H}$)
  Initialize $S \leftarrow \emptyset$. ▷ candidate set
  repeat
    Uniformly randomly pick $\mathbf{d} \in \mathbb{F}_2^n$.
    Construct $\mathbf{H_d}$ and $\mathbf{G_d} = \mathbf{H_d^T H_d}$
    for each vector $\mathbf{s}_i \in \ker(\mathbf{G_d})$ **do**
      if $\mathbf{s}_i$ passes certain property check **then** ▷ To be specified
        Add $\mathbf{s}_i$ to S.
      end if
    end for
  until some stopping criterion is met.
  return S
end procedure

Meta-Algorithm 2: The EXTRACTSECRET($\mathbf{H}$) procedure of Linearity Attack.

5.1 Linearity Attack

Classical attacks based on secret extraction aim to mimic the quantum behavior on certain candidate set S. Observe that given an IQP circuit represented by the binary matrix $\mathbf{H}$, a quantum prover can output a sample $\mathbf{x}$, which has the correlation function $\langle \mathcal{Z}_\mathbf{s} \rangle$ in the direction of $\mathbf{s}$ for every $\mathbf{s}$, even if it is not the secret of the verifier. If a classical prover can also generate samples that have the correct correlation with every $\mathbf{s}$, then he has the power to classically sample from an IQP circuit, which is implausible [25, 26]. However, he has the knowledge that the verifier will only check one secret. Therefore, a general attack strategy for him is to first reduce the set of candidate secrets from $\{0,1\}^n$ to a (polynomial-sized) subset S, and then generate samples that have the correct correlation with every vector in the candidate set.

Here, we discuss Linearity Attack, which is an instance of classical attacks based on secret extraction and generalizes the attack algorithm in Ref. [30]. It consists of two steps. First, it uses linear algebraic techniques to construct a candidate set S. Then, the prover calculates the correlation function for every vector in S, and outputs samples that have the correct correlation with those vectors.

5.1.1 Secret extraction

Overview. The secret extraction procedure in the Linearity Attack is presented in Meta-Algorithm 2, which is a generalized version of the procedure described in Ref. [30]. The algorithm begins by randomly selecting a vector $\mathbf{d}$ and eliminating rows in $\mathbf{H}$ that are orthogonal to $\mathbf{d}$, resulting in $\mathbf{H_d}$. Subsequently, the algorithm searches for vectors that satisfy certain property check in $\ker(\mathbf{G_d})$, where $\mathbf{G_d} = \mathbf{H_d^T H_d}$ represents the Gram matrix associated with $\mathbf{d}$. In what follows, we discuss some technical details and defer the analysis to Section 5.2.

Secret extraction in Kahanamoku-Meyer's attack. Meta-Algorithm 2 differs slightly from the approach described in Ref. [30]. In the original algorithm, the classical prover begins by constructing a matrix $\mathbf{M} \in \mathbb{F}_2^{l \times n}$ through linear combinations of rows in $\mathbf{H}$. Specifically, after sampling the vector $\mathbf{d}$, the classical prover proceeds to sample l random vectors $\mathbf{e}_1, \ldots, \mathbf{e}_l$. Then, the j-th row of $\mathbf{M}$ is defined by,

$$\mathbf{m}_j^T := \sum_{\substack{\mathbf{p}^T \in \text{row}(\mathbf{H}) \\ \mathbf{p}\cdot\mathbf{d} = \mathbf{p}\cdot\mathbf{e}_j = 1}} \mathbf{p}^T \,. \tag{5.1}$$

After that, the original algorithm searches for the vectors that can pass certain property check in $\ker(\mathbf{M})$ instead.

Our secret extraction algorithm is a generalization and simplification to the original approach. In Appendix G.1, we show that rows in $\mathbf{M}$ belong to the row space of $\mathbf{G_d}$. Therefore, to minimize the size of $\ker(\mathbf{M})$, one can simply set $\mathbf{M} = \mathbf{G_d}$, eliminating the need to sample the vectors $\mathbf{e}_1, \ldots, \mathbf{e}_l$.

Property check. Next, we discuss the property checks designed to determine whether a vector in $\ker(\mathbf{G_d})$ can serve as a potential secret or not. In the context of the Shepherd-Bremner construction targeted in Ref. [30], the property check is to check whether $\mathbf{s}_i$ in $\ker(\mathbf{M})$ corresponds to a quadratic-residue code or not. To accomplish this, the prover constructs $\mathbf{H}_{\mathbf{s}_i}$ for the vector $\mathbf{s}_i$ and performs what we refer to as the QRC check, examining whether $\mathbf{H}_{\mathbf{s}_i}$ generates a quadratic-residue code (with possible row reordering). However, determining whether a generator matrix generates a quadratic-residue code is a nontrivial task. Consequently, the algorithm in Ref. [30] attempts to achieve this by assessing the weight of the codewords in the code generated by $\mathbf{H}_{\mathbf{s}_i}$. In a quadratic-residue code, the weight of the codewords will be either 0 or 3 (mod 4). But still, there will be exponentially many codewords, and checking the weights of the basis vectors is not sufficient to ensure that all codewords have weight either 0 or 3 (mod 4). So in practice, the prover can only check a small number of the codewords.

For instances derived from the stabilizer construction, the prover will have less information about the code $\mathcal{C}_\mathbf{s}$; he only has the knowledge that this code has a large doubly-even subcode, as quantified by the rank of $\mathbf{G_s}$. Therefore, the property check for Meta-Algorithm 2 involves checking whether the rank of $\mathbf{H}_{\mathbf{s}_i}^T \mathbf{H}_{\mathbf{s}_i}$ falls below certain threshold and whether self-dual intersection $\mathcal{D}_{\mathbf{s}_i}$ is doubly-even. However, determining an appropriate threshold presents a challenge for the classical prover, who can generally only make guesses. If the chosen threshold is smaller than the rank of $\mathbf{G_s}$, then the secret extraction algorithm will miss the real secret, even if it lies within $\ker(\mathbf{G_d})$.

Stopping criteria. Lastly, various stopping criteria can be employed in the secret extraction procedure. One approach is to halt the procedure once a vector successfully passes the property check, as adopted in Ref. [30]. Alternatively, the procedure can be stopped after a specific number of repetitions or checks. In our implementation, we utilize a combination of these two criteria. If no vectors are able to pass the property check before the stopping criterion is reached, an empty candidate set S is returned, indicating a failed attack. Conversely, if the candidate set S is non-empty, the attack proceeds to the classical sampling step to generate classical samples.

5.1.2 Classical sampling

Classical sampling based on multiple candidate secrets is nontrivial. Mathematically, the problem is formulated as follows.

Problem 5.2. *Given an IQP circuit C and a candidate set $S = \{\mathbf{s}_1, \ldots, \mathbf{s}_t\}$, outputs a sample $\mathbf{x}$ so that*

$$\mathbb{E}[(-1)^{\mathbf{x}\cdot\mathbf{s}_i}] = \langle \mathcal{Z}_{\mathbf{s}_i} \rangle \ , \tag{5.2}$$

for $i = 1, \ldots, t$, where $\mathbb{E}[\cdot]$ is over the randomness of the algorithm.

Note that $\mathbb{E}[(-1)^{\mathbf{x}\cdot\mathbf{s}_i}]$ is the expectation value of Eq. (1.2). We may allow a polynomially-bounded additive error in the problem formulation, considering the inevitable shot noise due to finite samples. The complexity of this problem depends on various situations. To the best of our knowledge, we are not aware of an efficient classical algorithm that solves this problem in general. In Appendix G.2, we present two sampling algorithms that will work in some special cases. A sufficient condition for these two sampling algorithms to work is that the candidate set is an independent subset of $\{0,1\}^n$.

Naive sampling algorithm. In this work, we mainly focus on the case $|S| = 1$, in which case the problem is easy to solve, yet remains worth discussing. A naive sampling algorithm is as follows. To generate samples with the correct correlation on $\mathbf{s}$, one just needs to output samples that are orthogonal to the candidate vector $\mathbf{s}'$ with probability $\beta_{\mathbf{s}'} = (\langle \mathcal{Z}_{\mathbf{s}'} \rangle + 1)/2$ and otherwise with probability $1 - \beta_{\mathbf{s}'}$. One can prove that if the candidate secret from the ExtractSecret procedure is the real secret $\mathbf{s}$, then the generated samples using this strategy will have the correlation function approximately $\langle \mathcal{Z}_{\mathbf{s}} \rangle$ with the real secret. Otherwise, the correlation function with the real secret will be zero. We have the following lemma (see Appendix F for the proof).

Lemma 5.3. *Given a matrix $\mathbf{H}$ and two vectors $\mathbf{s} \neq \mathbf{s}'$, let $\langle \mathcal{Z}_{\mathbf{s}} \rangle$ and $\langle \mathcal{Z}_{\mathbf{s}'} \rangle$ be their corresponding correlation functions, as defined in Eq. (1.3). If a sample $\mathbf{x}$ is generated to be a vector orthogonal to $\mathbf{s}'$ with probability $\beta_{\mathbf{s}'} = (\langle \mathcal{Z}_{\mathbf{s}'} \rangle + 1)/2$ and otherwise with probability $1 - \beta_{\mathbf{s}'}$, then $\mathbb{E}[(-1)^{\mathbf{x}\cdot\mathbf{s}}] = 0$.*

The above lemma holds even if $\mathbf{H}\mathbf{s} = \mathbf{H}\mathbf{s}'$, in which case $\mathbf{s}$ and $\mathbf{s}'$ are said to be *equivalent secrets*. Equivalent secrets have the same non-orthogonal and redundant part, and the correlation functions $\langle \mathcal{Z}_{\mathbf{s}} \rangle$ and $\langle \mathcal{Z}_{\mathbf{s}'} \rangle$ are the same. It is clear that the number of equivalent secrets is given by $2^{n-\mathrm{rank}(\mathbf{H})}$, which will be 1 if $\mathbf{H}$ is of full column rank. When there are multiple equivalent secrets, it could be the case that the vector $\mathbf{s}'$ is returned by the secret extraction procedure, because it can also pass the property check, even if it is not the real secret itself. In this case, our previous classical sampling algorithm can only give samples with zero correlation function on the real secret $\mathbf{s}$, according to Lemma 5.3.

Sampling according to H. To address this issue, we propose a second classical sampling algorithm. Observe that linear combination of rows in $\mathbf{R}_{\mathbf{s}}$ gives vectors that are orthogonal to $\mathbf{s}$ and summation of an odd number of rows in $\mathbf{H}_{\mathbf{s}}$ gives vectors that are not orthogonal to $\mathbf{s}$. We denote the former set of vectors $\mathbb{S}_0(\mathbf{s})$ and the latter $\mathbb{S}_1(\mathbf{s})$. The identification of these sets relies on determining the submatrices $\mathbf{H}_{\mathbf{s}}$ and $\mathbf{R}_{\mathbf{s}}$. To achieve this, it suffices to find a vector $\mathbf{s}'$ that is equivalent to the real secret $\mathbf{s}$. Therefore, upon receiving the candidate secret $\mathbf{s}'$ from the secret extraction procedure, the classical prover proceeds by computing $\langle \mathcal{Z}_{\mathbf{s}'} \rangle$ and $\beta_{\mathbf{s}'}$, followed by identifying $\mathbb{S}_0(\mathbf{s}')$ and $\mathbb{S}_1(\mathbf{s}')$. A sample $\mathbf{x}$ is drawn from $\mathbb{S}_0(\mathbf{s}')$ with probability $\beta_{\mathbf{s}'}$ and from $\mathbb{S}_1(\mathbf{s}')$ with probability $1 - \beta_{\mathbf{s}'}$. If the vector $\mathbf{s}'$ is equivalent to $\mathbf{s}$, then this sampling algorithm will generate samples with the correct correlation function with respect to the real secret $\mathbf{s}$, as opposed to the naive sampling algorithm.

This also explains why we consider IQP matrices of full column rank in the stabilizer construction. If the classical prover is given an IQP matrix $\mathbf{H}$ that is not full-rank, he can always apply an invertible matrix $\mathbf{Q}$ so that $\mathbf{HQ} = (\mathbf{H}', \mathbf{0})$, where $\mathbf{H}'$ is of full column rank. Then, he runs the secret extraction algorithm on $\mathbf{H}'$. Once a candidate secret is found, he can use it to identify the corresponding $\mathbb{S}_0$ and $\mathbb{S}_1$ from the original matrix $\mathbf{H}$, as well as computing the correlation function. Finally, if the identification matches that of the real secret, then using the second classical sampling algorithm will allow him to pass the test.

5.2 Analysis

Here, we present analysis on the secret extraction of Linearity Attack.

Probability of sampling a good d. First, we have the following proposition.

Proposition 5.4. *Given an IQP matrix $\mathbf{H}$ and two vectors $\mathbf{d}$ and $\mathbf{s}$, we have $\mathbf{G_s d} = \mathbf{G_d s}$, where $\mathbf{G_s} = \mathbf{H_s}^T \mathbf{H_s}$ and $\mathbf{G_d} = \mathbf{H_d}^T \mathbf{H_d}$. Therefore, $\mathbf{s}$ lies in $\ker(\mathbf{G_d})$ if and only if $\mathbf{G_s d} = \mathbf{0}$, which happens with probability 2^{-g} over all choices of $\mathbf{d}$, where $g = \operatorname{rank}(\mathbf{G_s})$ is the rank of $\mathbf{G_s}$.*

The proof is given in Appendix G.3. This proposition tells us that if the random $\mathbf{d}$ does not satisfy $\mathbf{G_s d} = \mathbf{0}$, then the verifier's secret $\mathbf{s}$ will not lie in $\ker(\mathbf{G_d})$. In this case, Meta-Algorithm 2 will not be able to find the correct secret from the kernel of $\mathbf{G_d}$, and it has to be started over with a new $\mathbf{d}$.

If the correlation function with respect to the real secret has inverse polynomial scaling, i.e., $2^{-g/2} = \Omega(1/\operatorname{poly}(n))$, then the probability of sampling a good $\mathbf{d}$ is also large, which is $2^{-g} = \Omega(1/\operatorname{poly}(n))$. This might appear advantageous for the attacker. But note that a classical attack cannot determine whether the sampled $\mathbf{d}$ is good or not before he can find the real secret. In fact, he even cannot *definitively* determine whether a vector $\mathbf{s}_i$ in $\ker(\mathbf{G_d})$ that passes the property check is the real secret or not.

Size of $\ker(\mathbf{G_d})$. The next question is, how large is the size of $\ker(\mathbf{G_d})$. This is important because the steps before the property check takes $O(n^3)$ time, which comes from the Gaussian elimination used to solve the linear system to find the kernel of $\mathbf{G_d}$. However, for the property check, the prover will potentially need to check every vectors in $\ker(\mathbf{G_d})$, which takes time proportional to its size. It is important to note that checking the basis vectors of $\ker(\mathbf{G_d})$ is not sufficient to find the real secret $\mathbf{s}$, because the linearity structure is not preserved under taking the Gram matrix. Even if $\mathbf{s} \in \ker(\mathbf{G_d})$, the basis vectors of the kernel space can all have high ranks for their associated Gram matrices. Below, we give an expected lower bound for the size of $\ker(\mathbf{G_d})$, with the proof presented in Appendix G.4.

Theorem 5.5. *Given $(\mathbf{H}, \mathbf{s}) \in \mathcal{H}_{n,m,g}$, randomly sample a vector $\mathbf{d}$. Then, the size of $\ker(\mathbf{G_d})$ is greater than $2^{n-m/2}$ in expectation over the choice of $\mathbf{d}$.*

Therefore, the size of $\ker(\mathbf{G_d})$ is increased exponentially by increasing n. The increase of n can be achieved by adding column redundancy, i.e., adding more all-zeros columns in Eq. (4.1). But in the stabilizer construction, the column redundancy cannot be arbitrarily large. Recall that to make the IQP matrix $\mathbf{H}$ full rank, one needs to add at least $n - r$ redundant rows, where $r = \operatorname{rank}(\mathbf{H_s})$. If $\mathbf{H}$ is not full rank, then as we discussed in Section 5.1.2, the classical prover can always perform column operations to effectively reduce the number of columns n, and hence reduce the dimension of $\ker(\mathbf{G_d})$.

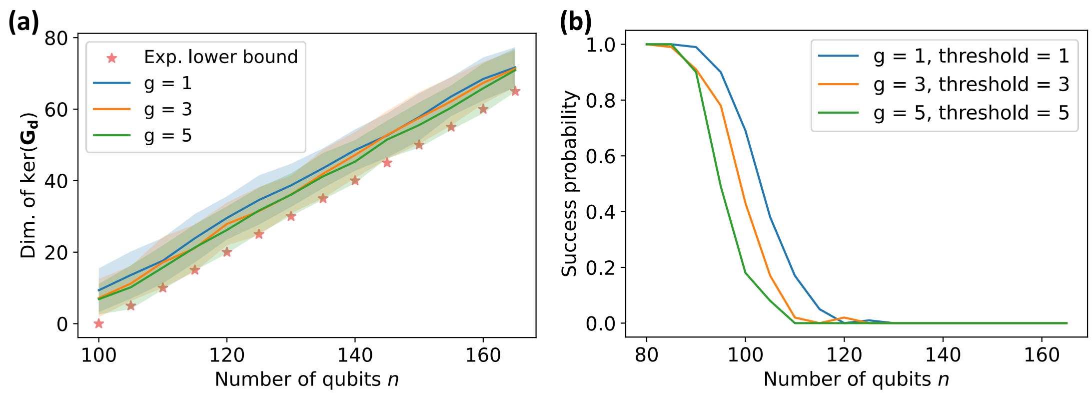

Figure 2: **(a)** The dimension of $\ker(\mathbf{G_d})$ for $g = 1, 3, 5$ and the number of rows $m = 200$. The asterisks indicate the expected lower bound $n - m/2$. **(b)** The success probability of the attack. Here, we set the threshold for the rank in the property check to be the same as g.

Suggested parameter regime. Based on the above analysis, it is important to choose a good parameter regime to invalidate the Linearity Attack. Suppose the expected security parameter is λ, meaning that the expected time complexity of a classical prover is $\Omega(2^\lambda)$. Then, generally we require $n - m/2 \geq \lambda$ for $\ker(\mathbf{G_d})$ to be sufficiently large, and the number of redundant rows $m - m_1 \geq n - r$ for $\mathbf{H}$ to be full-rank, where m_1 is the number of rows in $\mathbf{H_s}$. Specifically, for the stabilizer construction, given n and g, we randomly choose the parameter $r \geq g$. Then, we require that the number of rows in $\mathbf{H_s}$ and $\mathbf{H}$ satisfies

$$m_1 \leq n - 2\lambda + r \qquad\qquad m_1 + n - r \leq m \leq 2(n - \lambda)\,, \tag{5.3}$$

respectively. In addition, since m is the number of gates in the IQP circuit, we will require sufficiently large n and $m = \Omega(n)$ to invalidate classical simulation.

Numerical simulation. In Fig. 2 (a), we plot the dimension of $\ker(\mathbf{G_d})$ for $g = 1, 3, 5$ and $m = 200$. For each number of columns n, we sample 100 instances from $\mathcal{H}_{n,m,g}$ with the stabilizer construction (Meta-Algorithm 1). Then, a random $\mathbf{d}$ is sampled and we calculate the dimension of $\ker(\mathbf{G_d})$. The asterisks are the expected lower bound $n - m/2$, as shown in Theorem 5.5. The numerical experiment demonstrates good agreement with the theoretical prediction. In Fig. 2 (b), we present the numerical results for the success probability of the attack. Although to invalidate the attack, the maximum number of property checks should be 2^{50} or larger, we set it to be 2^{15} for a proof of principle in the numerical experiment. For each number of columns n, we sample 100 random instances from $\mathcal{H}_{n,m,g}$, where $m = 200$. Then, the Linearity Attack is applied to each instance and the success probability is defined as the fraction of successfully attacked instances, which is the instance that the attacker can classically generate samples to spoof the test. As one can see, the success probability decreases to zero as n exceeds $m/2 + 15 = 115$, as expected.

Challenge. In addition, we have posted a challenge problem as well as the source code for generation and verification on GitHub[1], to motivate further study. The challenge problem

[1] https://github.com/AlaricCheng/stabilizer_protocol_sim

is given by the $\mathbf{H}$ matrix of a random instance from $\mathcal{H}_{n,m,g}$ with $n = 300$ and $m = 360$; the g parameter is hidden because in practice, the prover can only guess a value. One needs to generate samples with the correct correlation function in the direction of the hidden secret to win the challenge.

5.3 A fix of the Shepherd-Bremner construction

Finally, we would like to remark why the attack in Ref. [30] can break the Shepherd-Bremner construction and how we can fix it by adding column redundancy. Let $\mathcal{H}^{\mathrm{QRC}}_{n,m,q} = \{(\mathbf{H}, \mathbf{s})\}$ be a family of pairs of an IQP matrix $\mathbf{H} \in \mathbb{F}_2^{m\times n}$ and a secret $\mathbf{s}$ so that $\mathbf{H}_\mathbf{s}$ generates a QRC of length q (up to row permutations) and $\mathbf{H}$ is of full column rank. What the construction recipe of Ref. [24] does is to randomly sample instances from $\mathcal{H}^{\mathrm{QRC}}_{n,m,q}$, where $n = (q+3)/2$ and $m \geq q$, leaving a loophole for the recent classical attack [30]. To see why the parameter regime is as above, we first note that the length of QRC is q, implying that the number of rows in $\mathbf{H}_\mathbf{s}$ is q and hence $m \geq q$. Moreover, the dimension of a length-q QRC is $(q+1)/2$, which implies that the rank of $\mathbf{H}_\mathbf{s}$ is $(q+1)/2$. But an all-ones column was added in the construction (see Eq. (2.4)), which is an codeword of QRC, leading to $n = (q+3)/2$.

In the Shepherd-Bremner construction, the rank of Gram matrix $\mathbf{G}_\mathbf{s}$ associated with the real secret $\mathbf{s}$ is 1 according to Corollary 3.3. Therefore, the probability of choosing a good $\mathbf{d}$ is $1/2$ (as also shown in Theorem 3.1 of Ref. [30]). However, since the number of columns and the number of rows in $\mathbf{H}$ is $n = (q+3)/2$ and $m \geq q$, respectively, the size of $\ker(\mathbf{G}_\mathbf{d})$ is generally small. As a result, the prover can efficiently explore the entire $\ker(\mathbf{G}_\mathbf{d})$, and if no vector passes the property check, the prover can simply regenerate $\mathbf{d}$ and repeat the secret extraction procedure. The numerical results in Ref. [30] indicated that the size of $\ker(\mathbf{G}_\mathbf{d})$ is indeed constant when applied to the Shepherd-Bremner construction, which suggests that an efficient classical prover can pass the test and hence break the original construction. Specifically, for the challenge instance posted in Ref. [24], m is taken to be $2q$. Then, according to Theorem 5.5, the dimension of $\ker(\mathbf{G}_\mathbf{d})$ is expected to be constant, making it susceptible to the attack.

To address this issue, the original Shepherd-Bremner construction can be enhanced by introducing additional column redundancy to extend the number of columns n, which can achieve random sampling from families $\mathcal{H}^{\mathrm{QRC}}_{n,m,q}$ with any $n \geq (q+1)/2$ (Appendix E). This hides the dimension information of the hidden QRC. Combined with other obfuscation techniques in the Shepherd-Bremner construction, this achieves random sampling from $\mathcal{H}^{\mathrm{QRC}}_{n,m,q}$ with any possible parameters.

Below, we propose a parameter regime that can invalidate the attack in Ref. [30]. Given the length q of the QRC, we have $r = (q+1)/2$ and $m_1 = q$ [29]. So, the first formula in Eq. (5.3) gives $n \geq (q-1)/2 + 2\lambda$ and the second formula gives the range of the number of redundant rows $n - (q+1)/2 \leq m_2 \leq 2n - 2\lambda - q$. In this way, the size of $\ker(\mathbf{G}_\mathbf{d})$ will be larger than 2^λ in general, offering a viable solution to fortify the Shepherd-Bremner construction against the attack. Note that the column redundancy technique was used in Ref. [28] to scramble a small random IQP circuit into a large one, to maintain the value of the correlation function, although its connection to the classical security was not explored. Moreover, a multi-secret version was explored in Ref. [41], which was shown to be more vulnerable to the classical attack instead.

We perform numerical experiment to support our previous analysis. When $m = 2q$, n can be as large as $r + q$ and the expected kernel dimension of $\mathbf{G}_\mathbf{d}$ is r. In Fig. 3 (a), we plot the kernel dimensions under the setting $n = r + q$ and $m = 2q$, with $q = 103, 127, 151$ and 167. For each

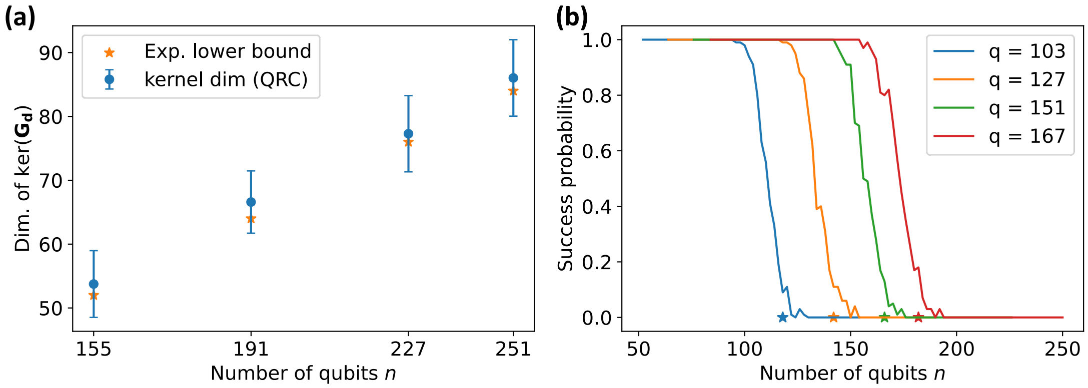

Figure 3: **(a)** The dimension of $\ker(\mathbf{G_d})$ for $q = 103, 127, 151, 167$. Here, the number of rows and columns are $m = 2q$ and $n = r + q$, where $r = (q+1)/2$ is the dimension of QRC. **(b)** The success probability of the attack. The asterisks denote the points $(q + 15, 0)$.

parameter set, 100 instances are sampled from $\mathcal{H}^{\mathrm{QRC}}_{n,m,q}$, and then a random $\mathbf{d}$ is sampled for each instance and we evaluate the dimension of $\ker(\mathbf{G_d})$. We also plot the expected lower bound $n - m/2$ for a comparison. In Fig. 3 (b), we plot the success probability versus the number of columns (qubits) n. Here, m is set to be $2q$ and n is increased from $r = (q+1)/2$ to $r + q$. For each value of n, 100 random instances from $\mathcal{H}^{\mathrm{QRC}}_{n,m,q}$ are sampled, and the success probability is the fraction of successful attacks among them. We set the security parameter to be 15 for a proof of principle, meaning that the maximum number of QRC checks is set to be 2^{15}. The success probabilities drop down to zero when $n > q + 15$, as expected. Our analysis and numerical results demonstrate that Claim 3.1 in Ref. [30], which originally states that the QRC-based construction can be broken efficiently by the KM attack, turns out to be false under appropriate choices of parameters.

6 Discussion

In this work, we give the stabilizer scheme for the IQP-based protocols for verifiable quantum advantage, which focuses on the case $\theta = \pi/8$ in the IQP circuits. With the connection between IQP circuits, stabilizer formalism and coding theory, we study the properties of correlation functions and IQP circuits. Based on these properties, we give an efficient procedure to sample generator matrices of random codes satisfying certain conditions, which lies at the core of our stabilizer scheme. Then, one needs to hide and obfuscate this generator matrix into a larger matrix. We propose a new obfuscation method called column redundancy, which uses the redundant generator matrix to hide the information of the dimension of the hidden code.

To explore the classical security of our protocol, we consider a family of attacks based on extracting secrets. We conjecture that such attacks cannot be efficient classically for random instances generated by our stabilizer scheme. To support this conjecture, we extend the recent attack algorithm on the QRC-based construction to the general case for $\theta = \pi/8$, which we call the Linearity Attack. Our analysis shows that this attack fails to find the secret in polynomial time by choosing instances from a good parameter regime. Notably, our column redundancy technique also fixes the loophole in the original Shepherd-Bremner construction. Our work paves the way for cryptographic verification of quantum computation advantage in the NISQ era.

There are several open problems for future research. The most important one is to rigorously prove the security of the IQP-based verification protocols. In Conjecture 5.1, we state that classical attacks based on secret extraction is on average hard. It would be favorable to prove the random self-reducibility of the problem, so that the hardness conjecture can be relaxed to the worst-case scenario. For example, recently a worst-to-average-case reduction was found for computing the probabilities of IQP circuits and it would be interesting to see if the techniques of Ref. [42] could be leveraged to gain insight into the validity of Conjecture 5.1. Before one can rigorously prove the hardness of classical attacks, one might gain intuition by considering other possible classical attacks. In terms of implementing the protocol in practice, generating instances according to a given architecture and noise analysis are also important open problems. We believe that the mathematical structure of the stabilizer scheme provides a promising avenue for the use of certain cryptographic techniques to improve the security of IQP-based protocols, and to construct instances that can be readily implemented with current technology.

Acknowledgement. We thank Ryan Snoyman for sharing his honors thesis, where he also considered the same problem and made some insightful observations. We also thank Earl Campbell, Ryan Mann, Mauro Morales and Man-Hong Yung for helpful discussions. BC acknowledges the support from the Sydney Quantum Academy. MJB acknowledges the support of Google. MJB acknowledges support by the ARC Centre of Excellence for Quantum Computation and Communication Technology (CQC2T), project number CE170100012. ZJ acknowledges the support of a startup funding from Tsinghua University.

References

[1] P.W. Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*. IEEE Comput. Soc. Press, 1994. 1

[2] Lov K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing - STOC '96*, pages 212–219, Philadelphia, Pennsylvania, United States, 1996. ACM Press. 1

[3] John Preskill. Quantum Computing in the NISQ era and beyond. *Quantum*, 2:79, August 2018. arXiv: 1801.00862. 1

[4] Sergio Boixo, Sergei V. Isakov, Vadim N. Smelyanskiy, Ryan Babbush, Nan Ding, Zhang Jiang, Michael J. Bremner, John M. Martinis, and Hartmut Neven. Characterizing quantum supremacy in near-term devices. *Nat. Phys.*, 14(6):595–600, June 2018. 1

[5] Frank Arute, Kunal Arya, Ryan Babbush, Dave Bacon, Joseph C. Bardin, Rami Barends, Rupak Biswas, Sergio Boixo, Fernando G. S. L. Brandao, David A. Buell, Brian Burkett, Yu Chen, Zijun Chen, Ben Chiaro, Roberto Collins, William Courtney, Andrew Dunsworth, Edward Farhi, Brooks Foxen, Austin Fowler, Craig Gidney, Marissa Giustina, Rob Graff, Keith Guerin, Steve Habegger, Matthew P. Harrigan, Michael J. Hartmann, Alan Ho, Markus Hoffmann, Trent Huang, Travis S. Humble, Sergei V. Isakov, Evan Jeffrey, Zhang

Jiang, Dvir Kafri, Kostyantyn Kechedzhi, Julian Kelly, Paul V. Klimov, Sergey Knysh, Alexander Korotkov, Fedor Kostritsa, David Landhuis, Mike Lindmark, Erik Lucero, Dmitry Lyakh, Salvatore Mandrà, Jarrod R. McClean, Matthew McEwen, Anthony Megrant, Xiao Mi, Kristel Michielsen, Masoud Mohseni, Josh Mutus, Ofer Naaman, Matthew Neeley, Charles Neill, Murphy Yuezhen Niu, Eric Ostby, Andre Petukhov, John C. Platt, Chris Quintana, Eleanor G. Rieffel, Pedram Roushan, Nicholas C. Rubin, Daniel Sank, Kevin J. Satzinger, Vadim Smelyanskiy, Kevin J. Sung, Matthew D. Trevithick, Amit Vainsencher, Benjamin Villalonga, Theodore White, Z. Jamie Yao, Ping Yeh, Adam Zalcman, Hartmut Neven, and John M. Martinis. Quantum supremacy using a programmable superconducting processor. *Nature*, 574(7779):505–510, October 2019. 1

[6] Qingling Zhu, Sirui Cao, Fusheng Chen, Ming-Cheng Chen, Xiawei Chen, Tung-Hsun Chung, Hui Deng, Yajie Du, Daojin Fan, Ming Gong, Cheng Guo, Chu Guo, Shaojun Guo, Lianchen Han, Linyin Hong, He-Liang Huang, Yong-Heng Huo, Liping Li, Na Li, Shaowei Li, Yuan Li, Futian Liang, Chun Lin, Jin Lin, Haoran Qian, Dan Qiao, Hao Rong, Hong Su, Lihua Sun, Liangyuan Wang, Shiyu Wang, Dachao Wu, Yulin Wu, Yu Xu, Kai Yan, Weifeng Yang, Yang Yang, Yangsen Ye, Jianghan Yin, Chong Ying, Jiale Yu, Chen Zha, Cha Zhang, Haibin Zhang, Kaili Zhang, Yiming Zhang, Han Zhao, Youwei Zhao, Liang Zhou, Chao-Yang Lu, Cheng-Zhi Peng, Xiaobo Zhu, and Jian-Wei Pan. Quantum computational advantage via 60-qubit 24-cycle random circuit sampling. *Science Bulletin*, page S2095927321006733, October 2021. 1

[7] Yulin Wu, Wan-Su Bao, Sirui Cao, Fusheng Chen, Ming-Cheng Chen, Xiawei Chen, Tung-Hsun Chung, Hui Deng, Yajie Du, Daojin Fan, Ming Gong, Cheng Guo, Chu Guo, Shaojun Guo, Lianchen Han, Linyin Hong, He-Liang Huang, Yong-Heng Huo, Liping Li, Na Li, Shaowei Li, Yuan Li, Futian Liang, Chun Lin, Jin Lin, Haoran Qian, Dan Qiao, Hao Rong, Hong Su, Lihua Sun, Liangyuan Wang, Shiyu Wang, Dachao Wu, Yu Xu, Kai Yan, Weifeng Yang, Yang Yang, Yangsen Ye, Jianghan Yin, Chong Ying, Jiale Yu, Chen Zha, Cha Zhang, Haibin Zhang, Kaili Zhang, Yiming Zhang, Han Zhao, Youwei Zhao, Liang Zhou, Qingling Zhu, Chao-Yang Lu, Cheng-Zhi Peng, Xiaobo Zhu, and Jian-Wei Pan. Strong Quantum Computational Advantage Using a Superconducting Quantum Processor. *Phys. Rev. Lett.*, 127(18):180501, October 2021. arXiv:2106.14734. 1

[8] Scott Aaronson and Alex Arkhipov. The computational complexity of linear optics. In *Proceedings of the Forty-Third Annual ACM Symposium on Theory of Computing*, STOC '11, page 333–342, New York, NY, USA, 2011. Association for Computing Machinery. 1

[9] Han-Sen Zhong, Hui Wang, Yu-Hao Deng, Ming-Cheng Chen, Li-Chao Peng, Yi-Han Luo, Jian Qin, Dian Wu, Xing Ding, Yi Hu, Peng Hu, Xiao-Yan Yang, Wei-Jun Zhang, Hao Li, Yuxuan Li, Xiao Jiang, Lin Gan, Guangwen Yang, Lixing You, Zhen Wang, Li Li, Nai-Le Liu, Chao-Yang Lu, and Jian-Wei Pan. Quantum computational advantage using photons. *Science*, 370(6523):1460–1463, December 2020. 1

[10] https://www.scottaaronson.com/blog/?p=284. At first sight, this seems a simple question. One may ask the quantum cloud to run a classical intractable task which is feasible for a quantum computer. This idea is not practical as it is equivalent to separating BQP (bounded-error quantum polynomial time) and P (polynomial time), one of the most important open problem in quantum complexity theory. 1

[11] A. Broadbent, J. Fitzsimons, and E. Kashefi. Universal blind quantum computation. In *2009 50th Annual IEEE Symposium on Foundations of Computer Science*, pages 517–526, Oct 2009. 1

[12] Anne Broadbent, Joseph Fitzsimons, and Elham Kashefi. Measurement-based and universal blind quantum computation. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6154 LNCS:43–86, 2010. 1

[13] Dorit Aharonov, Michael Ben-Or, and Elad Eban. Interactive Proofs For Quantum Computations, 2008. 1

[14] Dorit Aharonov, Michael Ben-Or, Elad Eban, and Urmila Mahadev. Interactive Proofs For Quantum Computations, 2008. 1

[15] Zhengfeng Ji. Classical verification of quantum proofs. In *Proceedings of the Forty-Eighth Annual ACM Symposium on Theory of Computing*, STOC '16, pages 885–898, New York, NY, USA, 2016. Association for Computing Machinery. 1

[16] Joseph F. Fitzsimons and Elham Kashefi. Unconditionally verifiable blind quantum computation. *Physical Review A*, 96(1), jul 2017. 1

[17] Joseph F. Fitzsimons, Michal Hajdušek, and Tomoyuki Morimae. Post hoc verification of quantum computation. *Phys. Rev. Lett.*, 120:040501, Jan 2018. 1

[18] Ben W. Reichardt, Falk Unger, and Umesh Vazirani. Classical command of quantum systems. *Nature*, 496(7446):456–460, 2013. 1

[19] Urmila Mahadev. Classical Verification of Quantum Computations. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 259–267, Paris, October 2018. IEEE. arXiv: 1804.01082. 1

[20] Zvika Brakerski, Paul Christiano, Urmila Mahadev, Umesh Vazirani, and Thomas Vidick. A Cryptographic Test of Quantumness and Certifiable Randomness from a Single Quantum Device. *arXiv:1804.00640 [quant-ph]*, April 2018. arXiv: 1804.00640. 1

[21] Zvika Brakerski, Venkata Koppula, Umesh Vazirani, and Thomas Vidick. Simpler Proofs of Quantumness. *arXiv:2005.04826*, May 2020. arXiv: 2005.04826. 1

[22] Gregory D. Kahanamoku-Meyer, Soonwon Choi, Umesh V. Vazirani, and Norman Y. Yao. Classically-Verifiable Quantum Advantage from a Computational Bell Test. *arXiv:2104.00687*, April 2021. arXiv: 2104.00687. 1

[23] Daiwei Zhu, Gregory D. Kahanamoku-Meyer, Laura Lewis, Crystal Noel, Or Katz, Bahaa Harraz, Qingfeng Wang, Andrew Risinger, Lei Feng, Debopriyo Biswas, Laird Egan, Alexandru Gheorghiu, Yunseong Nam, Thomas Vidick, Umesh Vazirani, Norman Y. Yao, Marko Cetina, and Christopher Monroe. Interactive cryptographic proofs of quantumness using mid-circuit measurements. *Nat. Phys.*, August 2023. arXiv:2112.05156. 1

[24] Dan Shepherd and Michael J Bremner. Temporally unstructured quantum computation. *Proc. R. Soc. A*, 465(2105):1413–1439, may 2009. 1, 1.1, 1.3, 2.4, 2.4, 5, 5.3, E

[25] Michael J. Bremner, Richard Jozsa, and Dan J. Shepherd. Classical simulation of commuting quantum computations implies collapse of the polynomial hierarchy. *Proc. R. Soc. A*, 467(2126):459–472, feb 2011. 1, 5.1

[26] Michael J. Bremner, Ashley Montanaro, and Dan J. Shepherd. Average-Case Complexity Versus Approximate Simulation of Commuting Quantum Computations. *Phys. Rev. Lett.*, 117(8):080501, August 2016. 1, 1.1, 1.3, 5.1

[27] Michael J Bremner, Ashley Montanaro, and Dan J Shepherd. Achieving quantum supremacy with sparse and noisy commuting quantum computations. *Quantum*, 1:8, 2017. 1

[28] Man-Hong Yung and Bin Cheng. Anti-Forging Quantum Data: Cryptographic Verification of Quantum Cloud Computing. *arXiv:2005.01510 [quant-ph]*, May 2020. arXiv: 2005.01510. 1, 1.1, 1.1, 1.3, 2.4, 3, 5.3, E

[29] Florence Jessie MacWilliams and Neil James Alexander Sloane. *The theory of error-correcting codes*. Elsevier, 1977. 1, 1.3, 5.3, C.3, D

[30] Gregory D. Kahanamoku-Meyer. Forging quantum data: classically defeating an IQP-based quantum test. *arXiv:1912.05547*, December 2019. arXiv: 1912.05547. 1, 1.2, 1.3, 5, 5.1, 5.1.1, 5.1.1, 5.1.1, 5.1.1, 5.3, 5.3, 3, G.1, G.2

[31] Dan Shepherd. Binary Matroids and Quantum Probability Distributions. *arXiv:1005.1744 [quant-ph]*, May 2010. arXiv: 1005.1744. 1, 1.1, 1.1, 1.2, 1.3, 2.4, 3

[32] Ryan L. Mann. Simulating quantum computations with Tutte polynomials. *npj Quantum Inf*, 7(1):141, September 2021. arXiv:2101.00211. 1, 1.3, 3

[33] Ref. [41] studies the multi-secret version of adding column redundancy and draws a conclusion that it makes the Shepherd-Bremner construction even more vulnerable. This is not in contradiction to our results, since we consider the single-secret version of adding column redundancy. 1

[34] Daniel Gottesman. The heisenberg representation of quantum computers. 1998. 1.1, 1.3

[35] Dirk Vertigan. Bicycle Dimension and Special Points of the Tutte Polynomial. *Journal of Combinatorial Theory, Series B*, 74(2):378–396, November 1998. 1.3, 3

[36] R. A. Pendavingh. On the evaluation at (-i,i) of the Tutte polynomial of a binary matroid. *J. Algebr. Comb.*, 39(1):141–152, February 2014. 1.3, 3

[37] Scott Aaronson and Daniel Gottesman. Improved Simulation of Stabilizer Circuits. *Phys. Rev. A*, 70(5):052328, November 2004. arXiv: quant-ph/0406196. 2.2, 2.1, A

[38] Xi Chen, Bin Cheng, Zhaokai Li, Xinfang Nie, Nengkun Yu, Man-Hong Yung, and Xinhua Peng. Experimental Cryptographic Verification for Near-Term Quantum Cloud Computing. *Sci. Bull.*, 66(1):23–28, 2021. arXiv: 1808.07375. 2.4

[39] Yong-Hyuk Kim and Keomkyo Seo. Two Congruence Classes for Symmetric Binary Matrices over F2. *WSEAS Trans. Math.*, 7(6):339–343, June 2008. 4, D

[40] Igor L. Markov and Yaoyun Shi. Simulating Quantum Computation by Contracting Tensor Networks. *SIAM J. Comput.*, 38(3):963–981, January 2008. arXiv: quant-ph/0511069. 5

[41] Ryan Snoyman. A Proof of Quantumness, 2020. Honor thesis, UNSW. 5.3, 33

[42] Ramis Movassagh. The hardness of random quantum circuits. *Nat. Phys.*, July 2023. 6

A Derivation of the overlap of two stabilizer states

Here, we would like to prove Proposition 2.1, which was discussed in Ref. [37]. Recall that given two stabilizer states $|\psi\rangle$ and $|\phi\rangle$, we have:

1. $\langle\psi|\phi\rangle = 0$ if their stabilizer groups contain the same Pauli operator of the opposite sign. That is, if there exists a Pauli operator P, such that $P \in \mathrm{Stab}(|\psi\rangle)$ and $-P \in \mathrm{Stab}(|\phi\rangle)$, then $\langle\psi|\phi\rangle = 0$.

2. Suppose the condition of the first case does not hold. Let g be the minimum number of different generators (i.e., the number of i s.t. $P_i \neq Q_i$). Then, $|\langle\psi|\phi\rangle| = 2^{-g/2}$.

Proof of Proposition 2.1. First, suppose $\mathrm{Stab}(|\psi\rangle) = \langle P_1, \cdots, P_n\rangle$ and $\mathrm{Stab}(|\phi\rangle) = \langle Q_1, \cdots, Q_n\rangle$. Then, we can write the states as,

$$|\psi\rangle\langle\psi| = \left(\frac{I+P_1}{2}\right)\cdots\left(\frac{I+P_n}{2}\right) \tag{A.1}$$

$$|\phi\rangle\langle\phi| = \left(\frac{I+Q_1}{2}\right)\cdots\left(\frac{I+Q_n}{2}\right) . \tag{A.2}$$

The square of the overlap is then given by,

$$|\langle\psi|\phi\rangle|^2 = \mathrm{Tr}(|\psi\rangle\langle\psi|\,|\phi\rangle\langle\phi|) = \mathrm{Tr}\left(\frac{I+P_1}{2}\cdots\frac{I+P_n}{2}\frac{I+Q_1}{2}\cdots\frac{I+Q_n}{2}\right) . \tag{A.3}$$

1. Suppose $Q_1 = -P_n$. Then, we have,

$$\frac{I+P_n}{2}\frac{I-P_n}{2} = 0 . \tag{A.4}$$

   Thus, $\langle\psi|\phi\rangle = 0$ in this case.

2. Suppose that $P_i = Q_i$ for $i > g$ and that the group $\langle P_1, \cdots, P_g\rangle$ is not equal to $\langle Q_1, \cdots, Q_g\rangle$. By commutation, we can group the same generators, which gives,

$$\frac{I+P_i}{2}\frac{I+Q_i}{2} = \frac{I+P_i}{2}\frac{I+P_i}{2} = \frac{I+P_i}{2} , \tag{A.5}$$

   for $i > g$. This will eliminate the terms related to $Q_{g+1}, \cdots, Q_n$. Then,

$$|\langle\psi|\phi\rangle|^2 = \mathrm{Tr}\left(\frac{I+P_1}{2}\cdots\frac{I+P_n}{2}\frac{I+Q_1}{2}\cdots\frac{I+Q_g}{2}\right) \tag{A.6}$$

$$= \frac{1}{2^g}\langle\psi|(I+Q_1)\cdots(I+Q_g)|\psi\rangle . \tag{A.7}$$

For every term $Q_iQ_j \cdots Q_k \neq I$ in the expansion, there exists a Pauli operator $P \in \mathrm{Stab}(|\psi\rangle)$ that anticommutes with it; otherwise, the term will be in the stabilizer group of $|\psi\rangle$. For such an operator Q, we have $\langle\psi|Q|\psi\rangle = 0$. Indeed, notice that

$$\langle\psi|Q|\psi\rangle = \langle\psi|QP|\psi\rangle = -\langle\psi|PQ|\psi\rangle = -\langle\psi|Q|\psi\rangle \,, \tag{A.8}$$

which implies $\langle\psi|Q|\psi\rangle = 0$. Finally, we have,

$$|\langle\psi|\phi\rangle|^2 = \frac{1}{2^g}\langle\psi|I|\psi\rangle = \frac{1}{2^g} \,, \tag{A.9}$$

and $|\langle\psi|\phi\rangle| = 2^{-g/2}$.

□

B Proofs for coding theory

We first prove the following proposition.

Proposition B.1. *The all-ones vector is a codeword of $\mathcal{C}$ if and only if its dual code $\mathcal{C}^\perp$ is an even code.*

Proof. Suppose $\mathbf{1} \in \mathcal{C}$. Then for every $\mathbf{c} \in \mathcal{C}^\perp$, we have $\mathbf{c} \cdot \mathbf{1} = 0$, which means that $|\mathbf{c}|$ is even and hence $\mathcal{C}^\perp$ is an even code. Conversely, suppose $\mathcal{C}^\perp$ is an even code. Then, all codewords will be orthogonal to the all-ones vector, and thus it is in $\mathcal{C}$. □

To prove that a weakly self-dual even code is either a doubly-even code or an unbiased even code, we will need the following lemma.

Lemma B.2. *Given two vectors $\mathbf{c}_1, \mathbf{c}_2 \in \mathbb{F}_2^m$ with even parity and $\mathbf{c}_1 \cdot \mathbf{c}_2 = 0$, let $\mathbf{c}_3 = \mathbf{c}_1 + \mathbf{c}_2$. Then, $|\mathbf{c}_3| = 0 \pmod 4$ if $|\mathbf{c}_1| = |\mathbf{c}_2| \pmod 4$ and $|\mathbf{c}_3| = 2 \pmod 4$ if $|\mathbf{c}_1| \neq |\mathbf{c}_2| \pmod 4$.*

Proof. Let $|\mathbf{c}_1| = a + 4k_1$ and $|\mathbf{c}_2| = b + 4k_2$, where $0 \leq a, b < 4$. Let the size of joint support of $\mathbf{c}_1$ and $\mathbf{c}_2$ be k_{12}. Then, $\mathbf{c}_1 \cdot \mathbf{c}_2 = k_{12} = 0 \pmod 2$, which means that k_{12} is an even number. So,

$$|\mathbf{c}_3| = a + b - 2k_{12} + 4(k_1 + k_2) = a + b \pmod 4 \,. \tag{B.1}$$

- If $|\mathbf{c}_1| = |\mathbf{c}_2| \pmod 4$, we have $a = b = 0$ or 2. In either case, $|\mathbf{c}_3| = 0 \pmod 4$.
- If $|\mathbf{c}_1| \neq |\mathbf{c}_2| \pmod 4$, we have $a = 0$ and $b = 2$ or $a = 2$ and $b = 0$. In either case, $|\mathbf{c}_3| = 2 \pmod 4$.

□

One can adapt the proof of this lemma to show that a doubly-even code is a weakly self-dual code. To see this, suppose $\mathcal{C}$ is a doubly-even code, and $\mathbf{c}_1, \mathbf{c}_2 \in \mathcal{C}$. Then, we have $|\mathbf{c}_1| = 4k_1$ and $|\mathbf{c}_2| = 4k_2$. Suppose $\mathbf{c}_3 = \mathbf{c}_1 + \mathbf{c}_2$, which gives $|\mathbf{c}_3| = 4(k_1 + k_2) - 2k_{12}$. Since $\mathbf{c}_3$ is also a codeword of a doubly-even code, we have $|\mathbf{c}_3| = 0 \pmod 4$, which implies that k_{12} is even and thus $\mathbf{c}_1 \cdot \mathbf{c}_2 = 0$.

Now, we are ready to prove Lemma 2.4.

Proof. Let $\mathcal{D}$ be a weakly self-dual even code spanned by $\{\mathbf{c}_1, \cdots, \mathbf{c}_d\}$. Then, $\mathbf{c}_i$'s are all even-parity and orthogonal to each other. Any codeword of $\mathcal{D}$ can be written as $\mathbf{c} = a_1\mathbf{c}_1 + \cdots + a_d\mathbf{c}_d$. According to Lemma B.2, in the linear combination of $\mathbf{c}$, if there is an odd number of $\mathbf{c}_i$'s with weight 2 modulo 4, then $\mathbf{c}$ will have weight 2 mod 4, and otherwise, $\mathbf{c}$ will have weight 0 mod 4. Therefore, if all $\mathbf{c}_i$'s have weight 0 modulo 4, then $\mathcal{D}$ is doubly-even. If there exist $\mathbf{c}_i$'s with weight 2 modulo 4, then $\mathcal{D}$ is an unbiased even code. □

Given a generator matrix $\mathbf{H}$, the rank of its Gram matrix $\mathbf{G} = \mathbf{H}^T\mathbf{H}$ is an invariant under a basis change. That is, $\mathrm{rank}(\mathbf{Q}^T\mathbf{G}\mathbf{Q}) = \mathrm{rank}(\mathbf{G})$ for $\mathbf{Q}$ invertible. It may be tentative to consider this as a direct consequence of Sylvester's law of inertia, but this is not the case since we are working in $\mathbb{F}_2$. Nevertheless, this can be proven as follows. First, the column space of $\mathbf{GQ}$ is a subspace of $\mathbf{G}$, which implies $\mathrm{rank}(\mathbf{GQ}) \leq \mathrm{rank}(\mathbf{G})$. On the other hand, the column space of $\mathbf{G}$ is a subspace of $\mathbf{GQ}$, because $\mathbf{GQQ}^{-1} = \mathbf{G}$, which implies $\mathrm{rank}(\mathbf{G}) \leq \mathrm{rank}(\mathbf{GQ})$. Therefore, we have $\mathrm{rank}(\mathbf{G}) = \mathrm{rank}(\mathbf{GQ})$. Applying this reasoning again gives $\mathrm{rank}(\mathbf{G}) = \mathrm{rank}(\mathbf{GQ}) = \mathrm{rank}(\mathbf{Q}^T\mathbf{G}\mathbf{G})$.

The rank of the Gram matrix measures how close a code $\mathcal{C}$ is to being a self-dual code. In particular, $\mathrm{rank}(\mathbf{G}) = \dim(\mathcal{C}) - \dim(\mathcal{D})$, where $\mathcal{D} := \mathcal{C} \bigcap \mathcal{C}^\perp$, which is the Proposition 2.5 in the main text.

Proof of Proposition 2.5. Suppose $\mathbf{H} \in \mathbb{F}_2^{m\times n}$ and let $g = \mathrm{rank}(\mathbf{G})$, where $\mathbf{G} = \mathbf{H}^T\mathbf{H}$. Let $r = \dim(\mathcal{C})$ and $d = \dim(\mathcal{D})$, where $d \leq r \leq n$. We first prove for the case $r = n$. In this case, every codeword in $\mathbf{c} \in \mathcal{C}$ can be expressed as $\mathbf{c} = \mathbf{Ha}$ for a unique $\mathbf{a} \in \mathbb{F}_2^n$ and the correspondence is one-to-one. Then, we claim that $\mathbf{Ha} \in \mathcal{D}$ is equivalent to $\mathbf{a} \in \ker(\mathbf{G})$ and thus d is equal to the dimension of $\ker(\mathbf{G})$, which is $d = r - g$. Indeed, if $\mathbf{Ha} \in \mathcal{D}$, we have $\mathbf{Ga} = \mathbf{H}^T\mathbf{Ha} = \mathbf{0}$, which means that $\mathbf{a} \in \ker(\mathbf{G})$. Conversely, if $\mathbf{a} \in \ker(\mathbf{G})$, we have $\mathbf{H}^T\mathbf{Ha} = \mathbf{0}$, which means that $\mathbf{Ha} \in \mathcal{C}^\perp$. Since $\mathbf{Ha} \in \mathcal{C}$, this implies $\mathbf{Ha} \in \mathcal{D}$.

Now, we consider the case $r < n$. In this case, there always exists an invertible matrix $\mathbf{Q}$ such that $\mathbf{HQ} = (\mathbf{H}', \mathbf{0}_{m\times(n-r)})$ and $\mathbf{H}' \in \mathbb{F}_2^{m\times r}$ is a generator matrix of $\mathcal{C}$ that is of full column rank. Moreover,

$$\mathrm{rank}(\mathbf{G}) = \mathrm{rank}(\mathbf{Q}^T\mathbf{G}\mathbf{Q}) = \mathrm{rank}(\mathbf{H}'^T\mathbf{H}') \,. \tag{B.2}$$

Then, applying the previous reasoning to $\mathbf{H}'$ yields that $d = r - \mathrm{rank}(\mathbf{H}'^T\mathbf{H}') = r - g$. □

C IQP circuits and stabilizer formalism

C.1 IQP stabilizer tableau

Theorem 3.1 can be proven in the following way. First, we start with the standard tableau of $|0^n\rangle$, which is

$$\begin{pmatrix} 0 & \dots & 0 & 1 & \dots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & \dots & 0 & 0 & \dots & 1 & 0 \end{pmatrix} , \tag{C.1}$$

corresponding to the stabilizer generators $\{Z_1, \cdots, Z_n\}$. Then, we apply the local terms in $e^{i\pi H/4}$ one by one, and keep track of the change of the stabilizer tableau. We have the following lemma which gives the form of Z_j conjugated by $e^{i\pi H/4}$.

Lemma C.1 (Evolution of Z_j)**.** *Let $\mathbf{H} = (\mathbf{c}_1, \mathbf{c}_2, \cdots, \mathbf{c}_n)$ be a binary matrix. Then, translating $\mathbf{H}$ into the IQP Hamiltonian H and after the conjugation of $e^{i\pi H/4}$, we have,*

$$e^{i\pi H/4} Z_j e^{-i\pi H/4} = i^{|\mathbf{c}_j|} \prod_{k=1}^{n} X_k^{\mathbf{c}_j \cdot \mathbf{c}_k} Z_j \,, \tag{C.2}$$

where $|\mathbf{c}_j|$ is the Hamming weight of $\mathbf{c}_j$.

For example, let

$$\mathbf{H} = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{pmatrix} \,. \tag{C.3}$$

Then, after the conjugation of $e^{i\pi H/4}$, we have

$$Z_1 \to (-1)(X_1X_2)(X_1X_4)Z_1 = -Z_1X_2X_4 \,. \tag{C.4}$$

Proof. First, note that $e^{i\pi H/4} = \prod_{\mathbf{p}^T \in \text{row}(\mathbf{H})} e^{i\pi \mathcal{X}_\mathbf{p}/4}$, where $\mathcal{X}_\mathbf{p} := X^{p_1} \otimes \cdots \otimes X^{p_n}$. For each row $\mathbf{p}^T$, if $p_j = 1$, then

$$e^{i\pi \mathcal{X}_\mathbf{p}/4} Z_j e^{-i\pi \mathcal{X}_\mathbf{p}/4} = e^{i\pi \mathcal{X}_\mathbf{p}/2} Z_j = i\mathcal{X}_\mathbf{p} Z_j \,; \tag{C.5}$$

and if $p_j = 0$, Z_j will remain unchanged. We suppose $p_j = 1$ for later illustration. Then, we apply the operator corresponding to another row $\mathbf{p}'^T$, which gives,

$$i e^{i\pi \mathcal{X}_{\mathbf{p}'}/4} \mathcal{X}_\mathbf{p} Z_j e^{-i\pi \mathcal{X}_{\mathbf{p}'}/4} = i\mathcal{X}_\mathbf{p} e^{i\pi \mathcal{X}_{\mathbf{p}'}/4} Z_j e^{-i\pi \mathcal{X}_{\mathbf{p}'}/4} \,. \tag{C.6}$$

If $p'_j = 1$, we have that the post-evolution stabilizer is given by $i^2 \mathcal{X}_\mathbf{p} \mathcal{X}_{\mathbf{p}'} Z_j$. In general, let $\mathbf{H}_j$ be the submatrix of $\mathbf{H}$ that consists of all rows whose j-th entry is 1. Then, after the conjugation of $e^{i\pi H/4}$, we have

$$e^{i\pi H/4} Z_j e^{-i\pi H/4} = i^{|\mathbf{c}_j|} \prod_{\mathbf{p}^T \in \text{row}(\mathbf{H}_j)} \mathcal{X}_\mathbf{p} Z_j \,. \tag{C.7}$$

For the Pauli X's in the above, whether there is the X_k component depends on the number of 1's in both the j-th and k-th column of $\mathbf{H}$. Indeed, the exponent of X_k is equal to $\mathbf{c}_j \cdot \mathbf{c}_k$. This completes the proof. □

Next, we are ready to prove Theorem 3.1.

Proof of Theorem 3.1. Since Z_j is the j-th stabilizer generator fo $|0^n\rangle$, Lemma C.1 actually gives the j-th stabilizer generator of $|\psi\rangle = e^{i\pi H/4} |0^n\rangle$. We can also write it in the following form,

$$(-1)^{r_j} \prod_{k=1}^{n} i^{\mathbf{c}_j \cdot \mathbf{c}_j} X_k^{\mathbf{c}_j \cdot \mathbf{c}_k} Z_j \;\;, \tag{C.8}$$

where $2r_j + \mathbf{c}_j \cdot \mathbf{c}_j = |\mathbf{c}_j| \pmod 4$ (note that the inner product is taken over $\mathbb{F}_2$). Therefore, if one uses $00, 01, 10, 11$ to represent $|\mathbf{c}_j| = 0, 1, 2, 3 \pmod 4$, then r_j is equal to the first bit. Finally, from this form of stabilizer generators, we can write down the stabilizer tableau of $|\psi\rangle$ as

$$\begin{pmatrix} \mathbf{c}_1 \cdot \mathbf{c}_1 & \dots & \mathbf{c}_1 \cdot \mathbf{c}_n & 1 & \dots & 0 & r_1 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{c}_n \cdot \mathbf{c}_1 & \dots & \mathbf{c}_n \cdot \mathbf{c}_n & 0 & \dots & 1 & r_n \end{pmatrix} \,. \tag{C.9}$$

□

C.2 Correlation function

Here, we prove Theorem 3.2 which connects the correlation function $\langle \mathcal{Z}_{\mathbf{s}} \rangle$, the code $\mathcal{C}_{\mathbf{s}}$ generated by $\mathbf{H}_{\mathbf{s}}$ and the rank of the Gram matrix $\mathbf{G} = \mathbf{H}_{\mathbf{s}}^T \mathbf{H}_{\mathbf{s}}$.

Proof of Theorem 3.2. First, $\mathbf{H}_{\mathbf{s}}\mathbf{s} = \mathbf{1}$ implies that $\mathcal{C}_{\mathbf{s}}^{\perp}$ is an even code and so is $\mathcal{D}_{\mathbf{s}}$. When $\theta = \pi/8$, we have

$$
\begin{aligned}
\langle \mathcal{Z}_{\mathbf{s}} \rangle &= \langle 0^n | e^{i2\theta H_{\mathbf{s}}} | 0^n \rangle & \text{(C.10)} \\
&= \langle 0^n | \prod_{\mathbf{p}^T \in \mathrm{row}(\mathbf{H}_{\mathbf{s}})} e^{i2\theta \mathcal{X}_{\mathbf{p}}} | 0^n \rangle & \text{(C.11)} \\
&= \langle 0^n | \prod_{\mathbf{p}^T \in \mathrm{row}(\mathbf{H}_{\mathbf{s}})} \frac{1}{\sqrt{2}} (I + i\mathcal{X}_{\mathbf{p}}) | 0^n \rangle & \text{(C.12)} \\
&= \frac{1}{\sqrt{2^m}} \sum_{\mathbf{a} \in \{0,1\}^m} i^{|\mathbf{a}|} \langle 0^n | \mathcal{X}_{\mathbf{a}^T \mathbf{H}_{\mathbf{s}}} | 0^n \rangle & \text{(C.13)} \\
&= \frac{1}{\sqrt{2^m}} \sum_{\mathbf{a} : \mathbf{a}^T \mathbf{H}_{\mathbf{s}} = \mathbf{0}} i^{|\mathbf{a}|} & \text{(C.14)} \\
&= \frac{1}{\sqrt{2^m}} \sum_{\mathbf{a} \in \mathcal{C}_{\mathbf{s}}^{\perp}} i^{|\mathbf{a}|} \,, & \text{(C.15)}
\end{aligned}
$$

where m is the number of rows in $\mathbf{H}_{\mathbf{s}}$. Since $\mathcal{C}_{\mathbf{s}}^{\perp}$ is an even code, we can write,

$$
\langle \mathcal{Z}_{\mathbf{s}} \rangle = \frac{1}{\sqrt{2^m}} \left(\sum_{\substack{\mathbf{a} \in \mathcal{C}_{\mathbf{s}}^{\perp} \\ |\mathbf{a}| = 0 \bmod 4}} 1 - \sum_{\substack{\mathbf{a} \in \mathcal{C}_{\mathbf{s}}^{\perp} \\ |\mathbf{a}| = 2 \bmod 4}} 1 \right) . \tag{C.16}
$$

Let $d = \dim(\mathcal{D}_{\mathbf{s}})$, $r = \dim(\mathcal{C}_{\mathbf{s}})$ and $g = r - d$.

One can always find an invertible matrix $\mathbf{Q}$, such that in $\mathbf{H}_{\mathbf{s}}\mathbf{Q}$, the first g columns are in $\mathcal{C}_{\mathbf{s}} \backslash \mathcal{D}_{\mathbf{s}}$, the g-th to the r-th columns form a basis of $\mathcal{D}_{\mathbf{s}}$ and the remaining columns are all-zeros. This transformation will not change the value of the correlation function according to Eq. (C.16), because it preserves the code $\mathcal{C}_{\mathbf{s}}$ and hence the dual code $\mathcal{C}_{\mathbf{s}}^{\perp}$. Under this transformation, the stabilizer tableau related to $\mathbf{H}_{\mathbf{s}}\mathbf{Q}$ is given by $(\mathbf{G}', \mathbf{I}_n, \mathbf{r}')$. Here, only the top-left $g \times g$ submatrix of $\mathbf{G}'$ can be nonzero, and all other entries are zero. According to Proposition 2.5, the rank of $\mathbf{G}$ is also g, which means that the $g \times g$ submatrix is full rank.

As for the phase column $\mathbf{r}'$, if the basis of $\mathcal{D}_{\mathbf{s}}$ have weight 0 modulo 4, then only the first g entries of $\mathbf{r}'$ can be nonzero, and all other entries are zero, according to Theorem 3.1. In this case, $\mathcal{D}_{\mathbf{s}}$ is a doubly-even code. For this set of generators represented by the transformed tableau, the number of non-Z generators is g, corresponding to the first g rows of $(\mathbf{G}', \mathbf{I}_n, \mathbf{r}')$. This is the minimum number over all possible choices, since the top-left submatrix of $\mathbf{G}'$ is already full rank. Thus, the correlation function is nonzero and has a magnitude $2^{-g/2}$, according to Proposition 2.1.

On the other hand, if in $\mathbf{H}_{\mathbf{s}}\mathbf{Q}$, some basis of $\mathcal{D}_{\mathbf{s}}$ have weight 2 modulo 4, then the corresponding entries in $\mathbf{r}'$ are 1, which gives Z-products with minus sign in the stabilizer group of $|\psi_{\mathbf{s}}\rangle := e^{i\pi H_{\mathbf{s}}/4} |0^n\rangle$. This means that $|\psi_{\mathbf{s}}\rangle$ has zero overlap with $|0^n\rangle$ and hence the correlation function is zero. In this case, it can be shown that $\mathcal{D}_{\mathbf{s}}$ is an unbiased even code using Lemma 2.4. $\square$

C.3 Stabilizer characterization applied to the Shepherd-Bremner construction

Here, we prove Corollary 3.3, which follows from Theorem 3.1 and the properties of QRC.

Proof. The rank of QRC is $(q+1)/2$, which means that there are $n=(q+3)/2$ columns in $\mathbf{H}_{\mathbf{s}}^{\mathrm{QRC}}$. To prove this corollary, it suffices to prove the following,

$$|\mathbf{c}_j| = 3 \pmod 4 \qquad\qquad \mathbf{c}_j \cdot \mathbf{c}_k = 1 \pmod 2\,, \tag{C.17}$$

according to Theorem 3.1.

First, the number of non-zero quadratic residues modulo q is $(q-1)/2$. Since $q+1$ is a multiple of 8, we have $|\mathbf{c}_j| = (q-1)/2 = 3 \pmod 4$ for $j \neq 1$. For $j=1$, $|\mathbf{c}_1| = q = 3 \pmod 4$.

As for the second formula, the cases (a) $j=k$, (b) $j=1$ but $k \neq 1$ and (c) $j \neq 1$ but $k=1$ follow the proof of the first formula. So, we focus on proving it for $j \neq k \neq 1$. Define the extended QRC by appending an extra parity bit to the codeword of QRC, which equals the Hamming weight of the codeword modulo 2. From classical coding theory, the extended QRC is self-dual [29]. That is, every two codewords of the extended QRC is orthogonal to each other. For $\mathbf{c}_j$, the added parity bit is 1, since these columns are odd-parity. Then, the fact that the extended codewords are orthogonal to each other implies that $\mathbf{c}_j \cdot \mathbf{c}_k = 1 \pmod 2$. This proves the form of the stabilizer tableau, which represents the generators $\{-Y_1X_2\cdots X_n, -X_1Y_2\cdots X_n, \cdots, -X_1X_2\cdots Y_n\}$.

Multiplying the first generator to the remaining $n-1$ generators gives the same stabilizer group with a different set of generators $\langle -Y_1X_2\cdots X_n, Z_1Z_2, Z_1Z_3\cdots, Z_1Z_n\rangle$. In this representation, the Z-type stabilizer generators have a positive phase and the number of non-Z generator is $g=1$. According to Proposition 2.1, the correlation function has a magnitude $1/\sqrt{2}$ (a.k.a. 0.854 probability bias) with respect to the secret, regardless of the size parameter q. □

D Details in stabilizer construction

We first give the following parameter constraints of the stabilizer construction, which are necessary conditions for instances from the family $\mathcal{H}_{n,m,g}$.

Proposition D.1 (Parameter constraints)**.** *Given $(\mathbf{H},\mathbf{s}) \in \mathcal{H}_{n,m,g}$, let $\mathcal{D}_{\mathbf{s}} = \mathcal{C}_{\mathbf{s}} \bigcap \mathcal{C}_{\mathbf{s}}^{\perp}$, where $\mathcal{C}_{\mathbf{s}}$ is the code generated by $\mathbf{H}_{\mathbf{s}}$ and $\mathcal{C}_{\mathbf{s}}^{\perp}$ is the dual code. Let m_1 be the number of rows in $\mathbf{H}_{\mathbf{s}}$ and $d = \dim(\mathcal{D}_{\mathbf{s}})$, which means $\dim(\mathcal{C}_{\mathbf{s}}) = g+d$. Then, we have*

- $g+d \leq n$;
- $0 < m_1 \leq m$;
- $n-g-d \leq m-m_1$;
- $g+2d \leq m_1$;
- $m_1 = g \bmod 2$.

Proof. The first constraint is from $\mathrm{rank}(\mathbf{H}_{\mathbf{s}}) \leq n$. The second one is trivial. The third is due to the fact that $\mathbf{H}$ is of full column rank, which means that the number of redundant rows should be $m-m_1 \geq n-\mathrm{rank}(\mathbf{H}_{\mathbf{s}}) = n-g-d$. The fourth is because $\dim(\mathcal{C}_{\mathbf{s}}) + \dim(\mathcal{C}_{\mathbf{s}}^{\perp}) = m_1$ and $\dim(\mathcal{D}_{\mathbf{s}}) \leq \dim(\mathcal{C}_{\mathbf{s}}^{\perp})$. The fifth one is from Theorem D.3 proved later. □

As stated in the main text, given the parameters m_1, n, d and g, the stabilizer construction is reduced to sampling a random $\mathbf{H_s}$ and $\mathbf{s}$, so that $\mathbf{H_s} = (\mathbf{F}, \mathbf{D}, \mathbf{0}_{m_1\times(n-r)})$, and $\mathbf{H_s s} = \mathbf{1}$, where $r = d + g$. Here, $\mathbf{D} \in \mathbb{F}_2^{m_1\times d}$ is a generator matrix of a random doubly-even code $\mathcal{D}_\mathbf{s} = \mathcal{C}_\mathbf{s} \bigcap \mathcal{C}_\mathbf{s}^\perp$, and $\mathbf{F} \in \mathbb{F}_2^{m_1\times g}$ span $\mathcal{C}_\mathbf{s}/\mathcal{D}_\mathbf{s}$ satisfying $\mathbf{D}^T\mathbf{F} = \mathbf{0}$ and $\mathrm{rank}(\mathbf{F}^T\mathbf{F}) = g$.

In more details, $\mathbf{D}$ and $\mathbf{F}$ shall satisfy the following conditions.

Proposition D.2 (Conditions of $\mathbf{D}$ and $\mathbf{F}$)**.** *Given $(\mathbf{H}, \mathbf{s}) \in \mathcal{H}_{n,m,g}$, let $\mathbf{H_s}$ be the rows of $\mathbf{H}$ that are not orthogonal to $\mathbf{s}$. Then, there exists an invertible $\mathbf{Q}$, so that $\mathbf{H_s Q} = (\mathbf{F}, \mathbf{D}, \mathbf{0})$ and*

- *$\mathbf{D}$ consists of $d = r - g$ independent vectors with weight 0 modulo 4, which are orthogonal to each other, with $r = \mathrm{rank}(\mathbf{H_s})$.*
- *$\mathbf{F}$ consists of g independent columns from $\ker(\mathbf{D}^T)$ which lie outside the column space of $\mathbf{D}$.*
- *$\mathbf{F}^T\mathbf{F}$ is a random full-rank g-by-g symmetric matrix.*
- *The all-ones vector $\mathbf{1}$ either explicitly appears as the first column of $\mathbf{D}$ or $\mathbf{F}$, or it can be written as the sum of the first two columns of $\mathbf{F}$.*

Proof. The matrix $\mathbf{D}$ is taken as the generator matrix of the dual intersection $\mathcal{D}_\mathbf{s}$, which is a doubly-even code. The form of $\mathbf{D}$ follows from Proposition 2.4. The second condition is because $\mathcal{D}_\mathbf{s} \subset \mathcal{C}_\mathbf{s}^\perp$, which implies $\mathbf{D}^T\mathbf{F} = \mathbf{0}$. So, columns of $\mathbf{F}$ lie in $\ker(\mathbf{D}^T)$. The third condition is because $\mathrm{rank}(\mathbf{F}^T\mathbf{F}) = \mathrm{rank}(\mathbf{Q}^T\mathbf{H}_\mathbf{s}^T\mathbf{H_s Q}) = \mathrm{rank}(\mathbf{H}_\mathbf{s}^T\mathbf{H_s}) = g$. As for the last condition, if $\mathbf{1} \in \mathcal{C}_\mathbf{s}$, one can always perform basis change so that $\mathbf{1}$ explicitly appears in the columns of $\mathbf{H_s}$. More specifically, if $\mathbf{1} \in \mathcal{D}_\mathbf{s}$, the column operation $\mathbf{Q}$ can transform it as the first column of $\mathbf{D}$. If not, it can be made as the first column of $\mathbf{F}$. The second part of this condition can be achieved by adding the second column of $\mathbf{F}$ to the first. □

Standard form. Although Ref. [39] gave the congruent standard form of any symmetric matrix over $\mathbb{F}_2$, for our purpose of ensuring the all-ones vector being a codeword, we consider a slightly different standard form. Such a standard form can be achieved by $\mathbf{H_s} = (\mathbf{F}, \mathbf{D}, \mathbf{0})$ where $\mathbf{F}$ and $\mathbf{D}$ satisfy the conditions in Proposition D.2. Then, the construction algorithm only needs to sample $\mathbf{H_s}$ so that $\mathbf{H}_\mathbf{s}^T\mathbf{H_s}$ is in the standard form. Note that $\mathbf{H_s}$ itself may not have a unique standard form under row permutations and column operations.

Theorem D.3. *Let $(\mathbf{H}, \mathbf{s})$ be a random instance from $\mathcal{H}_{n,m,g}$ and let $\mathbf{H_s}$ be the rows of $\mathbf{H}$ satisfying $\mathbf{H_s\,s} = \mathbf{1}$. Then, there exists an invertible matrix $\mathbf{Q}$, so that $\mathbf{H_s Q}$ is in the form of $(\mathbf{F}, \mathbf{D}, \mathbf{0})$ with $\mathbf{D}$ and $\mathbf{F}$ satisfying the conditions in Proposition D.2 and*

$$\mathbf{Q}^T\mathbf{H}_\mathbf{s}^T\mathbf{H_s Q} = \begin{pmatrix} \mathbf{I} & & & & \\ & \mathbf{J} & & & \\ & & \ddots & & \\ & & & \mathbf{J} & \\ & & & & \mathbf{0}_{(n-g)\times(n-g)} \end{pmatrix} \text{ or } \begin{pmatrix} \mathbf{J} & & & \\ & \ddots & & \\ & & \mathbf{J} & \\ & & & \mathbf{0}_{(n-g)\times(n-g)} \end{pmatrix}, \tag{D.1}$$

where $\mathbf{I}$ is either 1 or $\mathbf{I}_2$ and $\mathbf{J} := \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$. In addition, $m_1 = g \bmod 2$, where m_1 is the number of rows in $\mathbf{H_s}$.

Proof. First, according to Proposition D.2, $\mathbf{H_s}^T\mathbf{H_s} \sim_c \mathbf{F}^T\mathbf{F} \oplus \mathbf{D}^T\mathbf{D} \oplus \mathbf{0}$. For the $\mathbf{D}$ matrix, we have $\mathbf{D}^T\mathbf{D} = \mathbf{0}_{d\times d}$, which already matches standard form, where $d = \text{rank}(\mathbf{H_s}) - g$. So, we focus on $\mathbf{G}' := \mathbf{F}^T\mathbf{F}$. The matrix $\mathbf{F}$ satisfies $\mathbf{D}^T\mathbf{F} = \mathbf{0}$ and $\text{rank}(\mathbf{F}^T\mathbf{F}) = g$, where g is the number of columns in $\mathbf{F}$. We want to find an invertible matrix $\mathbf{Q}'$, which leaves the $\mathbf{D}$ matrix unchanged and only changes the $\mathbf{F}$ matrix, so that $\mathbf{G}'$ is equal to the top-left $g \times g$ submatrix in the standard form. We first discuss the congruent standard form of general full-rank symmetric matrix.

1. First, suppose that not all diagonal elements of $\mathbf{G}'$ are zero. In this case, we can assume $\mathbf{G}'_{11} = 1$, because otherwise, we can always apply a permutation matrix to $\mathbf{F}$, so that the nonzero diagonal element of $\mathbf{G}'$ is moved to the $(1,1)$-location. Then, up to congruent transformations,

$$\mathbf{G}' = \begin{pmatrix} 1 & \mathbf{g}^T \\ \mathbf{g} & \mathbf{G}_1 \end{pmatrix} . \tag{D.2}$$

   Let,

$$\mathbf{Q}_1 = \begin{pmatrix} 1 & \mathbf{g}^T \\ 0 & \mathbf{I} \end{pmatrix} . \tag{D.3}$$

   We have,

$$\mathbf{Q}_1^T\mathbf{G}'\mathbf{Q}_1 = \begin{pmatrix} 1 & \mathbf{0}^T \\ \mathbf{0} & \mathbf{g}\mathbf{g}^T + \mathbf{G}_1 \end{pmatrix} . \tag{D.4}$$

2. If $\mathbf{G}'_{jj} = 0$ for $1 \le j \le g$, then without loss of generality, we can assume $\mathbf{G}'_{12} = 1$; otherwise, we can apply a permutation matrix to swap the the non-zero entry to the $(1,2)$ and $(2,1)$ positions. In this case, up to congruent transformations,

$$\mathbf{G}' = \begin{pmatrix} \mathbf{J} & \mathbf{G}_2 \\ \mathbf{G}_2^T & \mathbf{G}_3 \end{pmatrix} , \tag{D.5}$$

   where $\mathbf{J} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$. Let

$$\mathbf{Q}_1 = \begin{pmatrix} \mathbf{I}_2 & \mathbf{J}\mathbf{G}_2 \\ \mathbf{0} & \mathbf{I}_{g-2} \end{pmatrix} . \tag{D.6}$$

   Then,

$$\mathbf{Q}_1^T\mathbf{G}'\mathbf{Q}_1 = \begin{pmatrix} \mathbf{J} & \mathbf{0}^T \\ \mathbf{0} & \mathbf{G}_2^T\mathbf{J}\mathbf{G}_2 + \mathbf{G}_3 \end{pmatrix} . \tag{D.7}$$

Therefore, we have $\mathbf{F}^T\mathbf{F} \sim_c (\mathbf{I}\oplus)\ \mathbf{J} \oplus \cdots \oplus \mathbf{J}$.

- If m_1 is odd, then $\mathbf{1}$ cannot be in $\mathcal{D}_\mathbf{s}$. In this case, $\mathbf{1}$ is the first column of $\mathbf{F}$, according to Proposition D.2. Then, $\mathbf{G}'_{11} = 1$ and applying the transformation of Eq. (D.3) leads to a matrix in the form of Eq. (D.4). This implies that all other columns in the new $\mathbf{F}$ are orthogonal to $\mathbf{1}$ and hence have even parity. Therefore, we have $\mathbf{G}'_{jj} = 0$ for $j > 1$ and $\mathbf{G}' \sim_c 1 \oplus \left(\bigoplus_{i=1}^{(g-1)/2} \mathbf{J} \right)$.

- If m_1 is even and $\mathbf{1} \notin \mathcal{D}_{\mathbf{s}}$, then $\mathbf{1}$ is also the first column of $\mathbf{F}$, according to Proposition D.2. Then, we can assume that $\mathbf{G}'_{12} = 1$, as what we did in proving the general congruent standard form. This implies that the second column of $\mathbf{F}$ must be odd-parity, and so $\mathbf{G}'_{22} = 1$. As as result, up to congruent transformations,

$$\mathbf{G}' = \begin{pmatrix} \mathbf{J}_1 & \mathbf{G}_2 \\ \mathbf{G}_2^T & \mathbf{G}_3 \end{pmatrix} , \tag{D.8}$$

where $\mathbf{J}_1 = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}$. Let

$$\mathbf{Q}_1 = \begin{pmatrix} \mathbf{I}_2 & \mathbf{J}_3\mathbf{G}_2 \\ \mathbf{0} & \mathbf{I}_{g-2} \end{pmatrix} , \tag{D.9}$$

where $\mathbf{J}_3 = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$. Then,

$$\mathbf{Q}_1^T\mathbf{G}'\mathbf{Q}_1 = \begin{pmatrix} \mathbf{J}_1 & \mathbf{0}^T \\ \mathbf{0} & \mathbf{G}_2^T\mathbf{J}_3\mathbf{G}_2 + \mathbf{G}_3 \end{pmatrix} . \tag{D.10}$$

From the second row (column) of $\mathbf{Q}_1^T\mathbf{G}'\mathbf{Q}_1$, one can see that only the second column of the new $\mathbf{F}$ is odd-parity and all other columns are even parity, which means that the diagonal elements of $\mathbf{G}_2^T\mathbf{J}_3\mathbf{G}_2 + \mathbf{G}_3$ are zero. Then, Eq. (D.6) is repeatedly applied, so that $\mathbf{G}' = \mathbf{J}_1 \oplus \mathbf{J} \oplus \cdots \oplus \mathbf{J}$. Finally, let $\mathbf{Q}_2$ be an invertible matrix that adds the second column to the first. We have $\mathbf{Q}_2^T\mathbf{G}'\mathbf{Q}_2 = \mathbf{I}_2 \oplus \mathbf{J} \oplus \cdots \oplus \mathbf{J}$.

- If m_1 is even and $\mathbf{1} \in \mathcal{D}_{\mathbf{s}}$, then m_1 must be a multiple of 4 and $\mathbf{1}$ is the first column of $\mathbf{D}$, according to Proposition D.2. In this case, $\mathbf{D}^T\mathbf{F} = \mathbf{0}$ implies that all columns of $\mathbf{F}$ will be even-parity, which means that the diagonal elements of $\mathbf{G}'$ will be zero. Moreover, the diagonal elements will remain zero if the congruent transformation only acts nontrivially on $\mathbf{G}'$. Therefore, $\mathbf{F}^T\mathbf{F} \sim_c \bigoplus_{i=1}^{g/2} \mathbf{J}$.

Above, the all-ones vector appears as the first column of $\mathbf{D}$ or $\mathbf{F}$ except for the third case, where $\mathbf{1} = \mathbf{c}_1 + \mathbf{c}_2$ can be obtained by adding up the first two columns of $\mathbf{F}$. Finally, in all of the above cases, $m_1 = g \bmod 2$. □

Next, we discuss the sampling of $\mathbf{D}$ and $\mathbf{F}$.

Sampling D. Here, the goal is to sample a $\mathbf{D} = (\mathbf{c}_1, \cdots, \mathbf{c}_d)$ with $d \leq m_1/2$, where $m_1 = g \bmod 2$. Columns in $\mathbf{D}$ are orthogonal to each other and have weight a multiple of 4, according to Proposition D.2. The algorithm is shown in Algorithm 1, which works as follows. First, $\mathbf{c}_1$ can be a random vector with weight 0 modulo 4; $\mathbf{D}$ is initialized as $\mathbf{D} = (\mathbf{c}_1)$. Then, the second column $\mathbf{c}_2$ is sampled with the constraint that $\mathbf{c}_1 \cdot \mathbf{c}_2 = 0$ and $|\mathbf{c}_2| = 0 \bmod 4$; $\mathbf{D}$ is updated to be $\mathbf{D} = (\mathbf{c}_1, \mathbf{c}_2)$. Next, the third column $\mathbf{c}_3$ is sampled so that it is orthogonal the first two columns and $|\mathbf{c}_3| = 0 \bmod 4$. This process is iterated until all d columns are sampled, or until no vector satisfying the condition can be sampled, in which case a matrix with $d - 1$ columns will be returned.

Parameters: m_1 and d
Require: $d \le m_1/2$
1: $\mathbf{c}_1 \leftarrow$ a random vector with weight 0 modulo 4
2: $\mathbf{D} \leftarrow (\mathbf{c}_1)$
3: **for** $t = 1, \cdots, d-1$ **do**
4: $\quad \mathbf{c}_{t+1} \leftarrow$ a random vector from $\ker(\mathbf{D}^T)/\langle \mathbf{c}_1, \cdots, \mathbf{c}_t \rangle$ with weight 0 modulo 4
5: $\quad$ **if** $\mathbf{c}_{t+1}$ does not exist **then**
6: $\qquad$ break
7: $\quad$ **end if**
8: $\quad \mathbf{D} \leftarrow (\mathbf{D}, \mathbf{c}_{t+1})$
9: **end for**
10: **if** $\mathbf{1}$ lies in the column space of $\mathbf{D}$ **then**
11: $\quad$ Apply column operations so that $\mathbf{1}$ is the first column of $\mathbf{D}$
12: **end if**
13: **return D**

Algorithm 1: Algorithm to sample a $\mathbf{D} = (\mathbf{c}_1, \mathbf{c}_2, \cdots, \mathbf{c}_d)$ so that $\mathbf{c}_i \cdot \mathbf{c}_j = 0$ and $|\mathbf{c}_i| = 0 \bmod 4$.

In the t-th iteration, the vector $\mathbf{c}_t$ is sampled from $\ker(\mathbf{D}^T)/\langle \mathbf{c}_1, \cdots, \mathbf{c}_{t-1} \rangle$ with $\mathbf{D} = (\mathbf{c}_1, \cdots, \mathbf{c}_{t-1})$. That is, we want $\mathbf{c}_t$ to be orthogonal to the first $t-1$ columns and outside the linear subspace that they span. This can be achieved as follows. We first solve for a basis of $\ker(\mathbf{D}^T)$, and then the first $t-1$ of the basis vectors are set as $\{\mathbf{c}_1, \cdots, \mathbf{c}_{t-1}\}$, with the remaining basis vectors changed accordingly. The vector $\mathbf{c}_t$ is sampled to be the random linear combination of the remaining basis vectors. In this way, the orthogonality and independence of $\mathbf{c}_t$ with respect to $\mathbf{c}_1, \cdots, \mathbf{c}_{t-1}$ are guaranteed.

In addition, to ensure that $|\mathbf{c}_t| = 0 \bmod 4$, we can first sample an even-parity vector from $\ker(\mathbf{D}^T)/\langle \mathbf{c}_1, \cdots, \mathbf{c}_{t-1} \rangle$. It is well-known that for a linear subspace over $\mathbb{F}_2$, either all vectors are even-parity or half the vectors are even-parity. Therefore, the sampling of even-parity vector can be efficiently done and we denote resulted vector as $\mathbf{a}_1$. The weight of $\mathbf{a}_1$ will be either 0 or 2 modulo 4. If $|\mathbf{a}_1| = 0 \bmod 4$, then it is set to be $\mathbf{c}_t$. Otherwise, we sample a vector $\mathbf{a}_2$ from $\ker(\mathbf{D}^T)/\langle \mathbf{c}_1, \cdots, \mathbf{c}_{t-1}, \mathbf{a}_1 \rangle$ that is orthogonal to $\mathbf{a}_1$; that is, $\mathbf{a}_2$ is a random vector from $\ker(\mathbf{D}^T)$ that is orthogonal to and outside $\langle \mathbf{c}_1, \cdots, \mathbf{c}_{t-1}, \mathbf{a}_1 \rangle$. Then, if $|\mathbf{a}_2| = 0 \bmod 4$, it is set to be $\mathbf{c}_t$ and if not, it follows from Lemma B.2 that $\mathbf{a}_1 + \mathbf{a}_2$ must have a weight that is a multiple of 4, and thus we assign it as $\mathbf{c}_t$. With this approach, a $\mathbf{c}_t$ with weight a multiple of 4 can be guaranteed to be sampled except for the final iteration of two extremal cases.

We now turn to discuss the cases $d = m_1/2$ and $d = (m_1 - 1)/2$, where in the last iteration, the column $\mathbf{c}_d$ *may* not exist. In such a case, only a matrix $\mathbf{D}$ with $d-1$ columns will be returned. However, we would like to emphasize that it is the g parameter that affects the value of the correlation function. For the subspace $\mathcal{D}_\mathbf{s}$, we only require it to be doubly-even, and its dimension does not matter. Therefore, we do not require the sampling algorithm succeed every time when applied to these two extremal cases.

When $d = m_1/2$ with m_1 even, the resulting $\mathcal{D}_\mathbf{s}$ forms a doubly-even self-dual code, which implies $g = 0$. In this scenario, $\mathbf{1}$ is included in $\mathcal{D}_\mathbf{s}$ because $\mathbf{1}$ will be orthogonal to all vectors in $\mathcal{D}_\mathbf{s}$ and itself. This implies that m_1 must be a multiple of 4. An example of this case is the

Input: m_1, g and $\mathbf{D}$
Require: $g \leq m_1 - 2d$ with d the number of columns in $\mathbf{D}$; $g = m_1 \bmod 2$

$\mathcal{D} \leftarrow$ column space of $\mathbf{D}$
if m_1 is odd **then** $\quad \triangleright\ \mathbf{G}' = \mathrm{diag}(1, \mathbf{J}, \cdots, \mathbf{J})$ and g is odd
  $\mathbf{c}_1 \leftarrow \mathbf{1}_{m_1}$
  $\mathbf{F} \leftarrow (\mathbf{c}_1)$
else if m_1 is even and $\mathbf{1}_{m_1} \notin \mathcal{D}$ **then** $\quad \triangleright\ \mathbf{G}' = \mathrm{diag}(\mathbf{I}_2, \mathbf{J}, \cdots, \mathbf{J})$
  $\mathbf{c}_2 \leftarrow$ a random odd-parity vector in $\ker(\mathbf{D}^T)/\mathcal{D}$
  $\mathbf{c}_1 \leftarrow \mathbf{1}_{m_1} + \mathbf{c}_2$
  $\mathbf{F} \leftarrow (\mathbf{c}_1, \mathbf{c}_2)$
else $\quad \triangleright\ \mathbf{G}' = \mathrm{diag}(\mathbf{J}, \cdots, \mathbf{J})$
  $\mathbf{c}_1 \leftarrow$ a random vector in $\ker(\mathbf{D}^T)/\mathcal{D}$
  $\mathbf{c}_2 \leftarrow$ a random vector in $\ker(\mathbf{D}^T)/\mathcal{D}$ that satisfies $\mathbf{c}_1 \cdot \mathbf{c}_2 = 1$
  $\mathbf{F} \leftarrow (\mathbf{c}_1, \mathbf{c}_2)$
end if
while number of columns in $\mathbf{F} < g$ **do**
  $\mathcal{C} \leftarrow \mathcal{D} \oplus$ column space of $\mathbf{F}$
  $\mathbf{a} \leftarrow$ a random vector in $\mathcal{C}^{\perp}/\mathcal{D}$
  $\mathbf{b} \leftarrow$ a random vector in $\mathcal{C}^{\perp}/\mathcal{D}$ that satisfies $\mathbf{a} \cdot \mathbf{b} = 1$
  $\mathbf{F} \leftarrow (\mathbf{F}, \mathbf{a}, \mathbf{b})$
end while
return F

Algorithm 2: Algorithm to sample $\mathbf{F} = (\mathbf{c}_1, \cdots, \mathbf{c}_g)$ so that $\mathbf{D}^T\mathbf{F} = \mathbf{0}$ and $\mathrm{rank}(\mathbf{F}^T\mathbf{F}) = g$.

extended QRC [29]. On the other hand, when $d = (m_1 - 1)/2$ with m_1 odd, we have $g = 1$ along with $\mathcal{D}_{\mathbf{s}} = \mathcal{C}_{\mathbf{s}}^{\perp}$. In this situation, the $\mathbf{F}$ matrix must be $\mathbf{1}$. The QRC serves as an example of this particular case.

The reason why the last iteration of Algorithm 1 may break on these two cases is as follows. We only discuss $d = m_1/2$, but the reasoning for $d = (m_1 - 1)/2$ is similar. When $t = d - 1$, $\mathbf{D} = (\mathbf{c}_1, \cdots, \mathbf{c}_{d-1})$. If $\mathbf{1}$ is not in $\langle \mathbf{c}_1, \cdots, \mathbf{c}_{d-1} \rangle$, then the algorithm will assign it as $\mathbf{c}_d$, and this iteration ends normally. However, if $\mathbf{1} \in \langle \mathbf{c}_1, \cdots, \mathbf{c}_{d-1} \rangle$, the dimension of $\ker(\mathbf{D}^T)$ is $m_1 - (d-1) = m_1/2 + 1$. So, the subspace that $\mathbf{a}_1$ is sampled from has dimension $m_1/2 + 1 - t = 2$. If this subspace has a nonzero vector with weight 0 modulo 4, then this vector will be assigned as $\mathbf{c}_d$ and the iteration will also end normally. However, if all vectors in this subspace have weight 2 modulo 4, except for the all-zeros vector, then $\mathbf{c}_d$ could not be found. In this case, the iteration breaks.

Sampling F. Next, we give the algorithm to sample $\mathbf{F} = (\mathbf{c}_1, \cdots, \mathbf{c}_g)$ (Algorithm 2). The algorithm takes m_1, g and $\mathbf{D}$ as inputs, and outputs a matrix $\mathbf{F}$ so that $\mathbf{F}^T\mathbf{F}$ is a rank-g symmetric matrix in the standard form as in Eq. (D.1), $\mathbf{D}^T\mathbf{F} = \mathbf{0}$, and $\mathbf{1}$ lies in the span of columns in $\mathbf{D}$ and $\mathbf{F}$. This implies that all columns of $\mathbf{F}$ should be sampled from $\ker(\mathbf{D}^T)/\mathcal{D}_{\mathbf{s}}$, with the additional orthogonality constraints imposed by $\mathbf{F}^T\mathbf{F}$.

There are three cases for $\mathbf{F}^T\mathbf{F}$. First, if m_1 is odd, then $\mathbf{1}$ cannot lie in $\mathcal{D}_{\mathbf{s}}$. According to

Proposition D.2 and Theorem D.3, $\mathbf{1}$ can be set as the first column of $\mathbf{F}$ and $\mathbf{F}^T\mathbf{F} = \text{diag}(1, \mathbf{J}, \cdots, \mathbf{J})$. Second, if m_1 is even but $\mathbf{1}$ is not in $\mathcal{D}_\mathbf{s}$, then $\mathbf{F}^T\mathbf{F} = \text{diag}(\mathbf{I}_2, \mathbf{J}, \cdots, \mathbf{J})$, according to Theorem D.3. In this case, $\mathbf{c}_1$ and $\mathbf{c}_2$ are odd-parity vectors, and $\mathbf{c}_1 + \mathbf{c}_2 = \mathbf{1}$. Third, if m_1 is even and $\mathbf{1}$ lies in $\mathcal{D}_\mathbf{s}$, then $\mathbf{F}^T\mathbf{F} = \text{diag}(\mathbf{J}, \cdots, \mathbf{J})$. In this case, $\mathbf{c}_1$ is a random vector from $\ker(\mathbf{D}^T)/\mathcal{D}_\mathbf{s}$ and $\mathbf{c}_2$ is a random vector from $\ker(\mathbf{D}^T)/\mathcal{D}_\mathbf{s}$ satisfying $\mathbf{c}_2 \cdot \mathbf{c}_1 = 1$. Note that $\mathbf{c}_1$ and $\mathbf{c}_2$ are automatically even-parity, since they are orthogonal to $\mathbf{1}$. Moreover, $\mathbf{c}_2$ must lie outside the space $\mathcal{D}_\mathbf{s} \oplus \langle \mathbf{c}_1 \rangle$, due to the constraint $\mathbf{c}_2 \cdot \mathbf{c}_1 = 1$.

After the initialization of $\mathbf{c}_1$ (and $\mathbf{c}_2$), the algorithm proceeds to sample other columns of $\mathbf{F}$, if $g > 1$ (or $g > 2$). We only illustrate the case when m_1 is odd below, but the sampling process for an even m_1 follows a similar pattern. For m_1 odd, $\mathbf{F}^T\mathbf{F} = \text{diag}(1, \mathbf{J}, \cdots, \mathbf{J})$. We first initialize $\mathcal{C}_\mathbf{s} \leftarrow \mathcal{D}_\mathbf{s} \oplus \langle \mathbf{c}_1 \rangle$, which is the subspace $\mathcal{C}$ in Algorithm 2. The block diagonal form of $\mathbf{F}^T\mathbf{F}$ implies that $\mathbf{c}_2$ and $\mathbf{c}_3$ are vectors from $\ker(\mathbf{D}^T)/\mathcal{D}_\mathbf{s}$ that are orthogonal to $\mathbf{c}_1 = \mathbf{1}$, i.e., $\mathbf{c}_2, \mathbf{c}_3 \in \mathcal{C}_\mathbf{s}^\perp/\mathcal{D}_\mathbf{s}$. For $\mathbf{c}_2$, it is sampled as a random vector from $\mathcal{C}_\mathbf{s}^\perp/\mathcal{D}_\mathbf{s}$, which is the vector $\mathbf{a}$ in Algorithm 2. For $\mathbf{c}_3$, it is sampled as a random vector from $\mathcal{C}_\mathbf{s}^\perp/\mathcal{D}_\mathbf{s}$ satisfying $\mathbf{c}_2 \cdot \mathbf{c}_3 = 1$, which is the vector $\mathbf{b}$ in Algorithm 2. This finishes the sampling of columns corresponding to the first $\mathbf{J}$ block. Then, the subspace $\mathcal{C}_\mathbf{s}$ is updated to include $\mathbf{c}_2$ and $\mathbf{c}_3$ into its basis, with its dimension increased by 2. This process is repeated for other columns, until all g columns are sampled. Finally, $\mathbf{F} = (\mathbf{c}_1, \cdots, \mathbf{c}_g)$ and it can be verified that $\mathbf{F}^T\mathbf{F}$ is indeed equal to the standard form $\text{diag}(1, \mathbf{J}, \cdots, \mathbf{J})$.

E Column redundancy

Essentially, adding column redundancy is to replace a full rank generator matrix of a code with a "redundant" generator matrix. The procedure of adding column redundancy is as follows.

1. Given a full-rank $\mathbf{H}_\mathbf{s}$, (e.g., the last 4 columns in Eq. (2.4)) and the secret, we first append all-zeros columns to $\mathbf{H}_\mathbf{s}$ and extend $\mathbf{s}$ accordingly,
$$\mathbf{H}_\mathbf{s} \leftarrow (\mathbf{H}_\mathbf{s}, \mathbf{0}) \qquad\qquad \mathbf{s} \leftarrow \begin{pmatrix} \mathbf{s} \\ \mathbf{s}' \end{pmatrix} . \tag{E.1}$$
2. Apply random column operations $\mathbf{Q}$ to obtain $\mathbf{H}_\mathbf{s} \leftarrow \mathbf{H}_\mathbf{s}\mathbf{Q}$ and $\mathbf{s} \leftarrow \mathbf{Q}^{-1}\mathbf{s}$.

Here, in the first step $\mathbf{s}'$ is an arbitrary vector whose length is the same as the number of all-zeros columns appended to $\mathbf{H}_\mathbf{s}$. Since the correlation function only depends on the linear code generated by $\mathbf{H}_\mathbf{s}$ [24, 28] and adding column redundancy does not change the linear code, the correlation function with respect to the new secret is unchanged after the above two steps. We would like to remark that although there are 2^{n_2} choices for $\mathbf{s}'$ of length n_2, once we fix a choice, the only constraint to the redundant rows is to be orthogonal to the specific new secret $\mathbf{s}$. Moreover, since the final IQP matrix $\mathbf{H}$ is of full column rank, only the real secret $\mathbf{s}$ will correspond to the code generated by $\mathbf{H}_\mathbf{s}$. In the case of QRC-based construction, if one chooses a redundant generator matrix of QRC by adding column redundancy, then n can be any integer larger than $(q+1)/2$, the dimension of QRC.

F Classical sampling and equivalent secrets

Here, we show that if a classical prover finds a wrong secret $\mathbf{s}'$, and generate classical samples using the naive sampling algorithm, then the generated samples will have the wrong correlation function on the real secret. We first prove the following proposition.

Proposition F.1. *For a nonzero* $\mathbf{s} \neq \mathbf{1}$*, if we randomly sample a vector* $\mathbf{d}$ *of even parity, then*

$$\Pr_{|\mathbf{d}|\ even}(\mathbf{s} \cdot \mathbf{d} = 1) = \Pr_{|\mathbf{d}|\ even}(\mathbf{s} \cdot \mathbf{d} = 0) = \frac{1}{2}\,. \tag{F.1}$$

Proof. The set of even-parity vectors forms a linear subspace. It is well known that for a linear subspace and a vector $\mathbf{s}$, either half of vectors are orthogonal to $\mathbf{s}$, or all vectors are orthogonal to $\mathbf{s}$. Since $\mathbf{s} \neq \mathbf{1}$, we have the former case. □

Given $\mathbf{H} = \begin{pmatrix} \mathbf{H_s} \\ \mathbf{R_s} \end{pmatrix}$ and $\mathbf{s}$, we say another vector $\mathbf{s}'$ is equivalent to $\mathbf{s}$ if $\mathbf{Hs} = \mathbf{Hs}'$; that is, they have the same inner-product relations with rows in $\mathbf{H}$. The following lemma shows that a random row orthogonal to $\mathbf{s}'$ will have probability $1/2$ to have inner product 1 with $\mathbf{s}$, even if $\mathbf{Hs} = \mathbf{Hs}'$.

Lemma F.2. *For* $\mathbf{s}' \neq \mathbf{s}$*, if we uniformly randomly sample a vector* $\mathbf{p}$ *orthogonal to* $\mathbf{s}'$*, then*

$$\Pr_{\mathbf{p}\cdot\mathbf{s}'=0}(\mathbf{p} \cdot \mathbf{s} = \mathbf{1}) = \frac{1}{2}\,. \tag{F.2}$$

Proof. Without loss of generality, assume $\mathbf{s}'$ has ones in the first k entries and zeros elsewhere. We can split $\mathbf{p} = \begin{pmatrix} \mathbf{p}_1 \\ \mathbf{p}_2 \end{pmatrix}$ and $\mathbf{s} = \begin{pmatrix} \mathbf{s}_1 \\ \mathbf{s}_2 \end{pmatrix}$, where $\mathbf{p}_1$ is a random even-parity string and $\mathbf{p}_2$ is uniformly random over $\mathbb{F}_2^{n-k}$. Then, $\mathbf{p} \cdot \mathbf{s} = \mathbf{p}_1 \cdot \mathbf{s}_1 + \mathbf{p}_2 \cdot \mathbf{s}_2$.

1. If $\mathbf{s}_2 = \mathbf{0}$ and $\mathbf{s}_1 \neq \mathbf{1}$,

$$\Pr_{\mathbf{p}\cdot\mathbf{s}'=0}(\mathbf{p} \cdot \mathbf{s} = \mathbf{1}) = \Pr_{\mathbf{p}_1\ \text{even}}(\mathbf{p}_1 \cdot \mathbf{s}_1 = \mathbf{1}) = \frac{1}{2}\,, \tag{F.3}$$

   according to Proposition F.1.

2. If $\mathbf{s}_2 \neq \mathbf{0}$ and $\mathbf{s}_1 = \mathbf{1}$,

$$\Pr_{\mathbf{p}\cdot\mathbf{s}'=0}(\mathbf{p} \cdot \mathbf{s} = \mathbf{1}) = \Pr_{\mathbf{p}_2}(\mathbf{p}_2 \cdot \mathbf{s}_2 = \mathbf{1}) = \frac{1}{2}\,, \tag{F.4}$$

   because $\mathbf{p}_2$ is uniformly random.

3. If $\mathbf{s}_2 \neq \mathbf{0}$ and $\mathbf{s}_1 \neq \mathbf{1}$,

$$\Pr_{\mathbf{p}\cdot\mathbf{s}'=0}(\mathbf{p} \cdot \mathbf{s} = \mathbf{1}) = \Pr_{\mathbf{p}_1,\mathbf{p}_2}(\mathbf{p}_1 \cdot \mathbf{s}_1 = \mathbf{1}, \mathbf{p}_2 \cdot \mathbf{s}_2 = \mathbf{0}) + \Pr_{\mathbf{p}_1,\mathbf{p}_2}(\mathbf{p}_1 \cdot \mathbf{s}_1 = \mathbf{0}, \mathbf{p}_2 \cdot \mathbf{s}_2 = \mathbf{1}) \tag{F.5}$$

$$= \frac{1}{2} \cdot \frac{1}{2} + \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{2}\,, \tag{F.6}$$

   where we used the independence of $\mathbf{p}_1$ and $\mathbf{p}_2$.

□

Now, we are ready to prove Lemma 5.3.

Parameter: number of linear equations l

```
procedure ExtractSecret(H)
    Uniformly randomly pick d ∈ F_2^n.
    for j = 1, 2, ··· , l do                      ▷ construct the linear-system matrix M
        Uniformly randomly pick e_j ∈ F_2^n.
        m_j^T ← RowSum(H_{d,e_j}).
    end for
    M ← (m_1, ··· , m_l)^T.
    for each vector s_i ∈ ker(M) do
        if s_i passes the QRC check then           ▷ discussed in the main text
            return s_i
        end if
    end for
end procedure
```

Meta-Algorithm 3: The ExtractSecret($\mathbf{H}$) subroutine in Ref. [30]. Here, given $\mathbf{H}$ and two vectors $\mathbf{d}$ and $\mathbf{e}_t$, we define $\mathbf{H}_{\mathbf{d},\mathbf{e}_j}$ to be a submatrix from $\mathbf{H}$ by deleting rows orthogonal to either $\mathbf{d}$ or $\mathbf{e}_j$.

Proof. With similar derivations to Lemma F.2, one can show that

$$\Pr_{\mathbf{p}\cdot\mathbf{s}'=0}(\mathbf{p}\cdot\mathbf{s}=\mathbf{0}) = \Pr_{\mathbf{p}\cdot\mathbf{s}'=1}(\mathbf{p}\cdot\mathbf{s}=\mathbf{0}) = \Pr_{\mathbf{p}\cdot\mathbf{s}'=1}(\mathbf{p}\cdot\mathbf{s}=\mathbf{1}) = \frac{1}{2}\,. \tag{F.7}$$

That means, if one samples a random $\mathbf{p}$ to be orthogonal to $\mathbf{s}'$ with probability β, and not orthogonal to $\mathbf{s}'$ with probability $1-\beta$, then

$$\Pr_{\mathbf{p}}(\mathbf{p}\cdot\mathbf{s}=0) = \beta \Pr_{\mathbf{p}\cdot\mathbf{s}=0}(\mathbf{p}\cdot\mathbf{s}=0) + (1-\beta)\Pr_{\mathbf{p}\cdot\mathbf{s}=1}(\mathbf{p}\cdot\mathbf{s}=0) = \frac{1}{2}\,. \tag{F.8}$$

That is, $\mathbf{p}$ is uncorrelated with $\mathbf{s}$ and the correlation function is

$$\mathbb{E}[(-1)^{\mathbf{p}\cdot\mathbf{s}}] = \Pr_{\mathbf{p}}(\mathbf{p}\cdot\mathbf{s}=0) - \Pr_{\mathbf{p}}(\mathbf{p}\cdot\mathbf{s}=1) = 0\,. \tag{F.9}$$

Therefore, if the secret extraction procedure returns a vector $\mathbf{s}' \neq \mathbf{s}$ and the classical prover uses the naive classical sampling algorithm to generate samples, then the samples will produce zero correlation function on the real secret. □

G More on Linearity Attack

G.1 Secret extraction in Kahanamoku-Meyer's attack

Meta-Algorithm 3 presents the secret extraction procedure in Ref. [30]. The procedure will be repeated if no vectors pass the QRC check. Here, we prove the following proposition.

Proposition G.1. *The matrix* $\mathbf{M}$ *obtained from Meta-Algorithm 3 consists of rows from the row space of* $\mathbf{G}_{\mathbf{d}} = \mathbf{H}_{\mathbf{d}}^T\mathbf{H}_{\mathbf{d}}$.

From this proposition, it is clear that to minimize the size of $\ker(\mathbf{M})$, one can choose $\mathbf{M} = \mathbf{G_d}$. In this way, the sampling of $\mathbf{e}_j$'s can be removed.

Proof. Recall that the j-th row of $\mathbf{M}$ is obtained in the following way. First, we eliminate rows in $\mathbf{H}$ that are orthogonal to $\mathbf{d}$, which gives $\mathbf{H_d}$. Then, we eliminate rows in $\mathbf{H_d}$ that are orthogonal to $\mathbf{e}_j$, which gives $\mathbf{H}_{\mathbf{d},\mathbf{e}_j}$. Finally, we sum up the rows in $\mathbf{H}_{\mathbf{d},\mathbf{e}_j}$, which gives $\mathbf{m}_j^T$. Equivalently, we have

$$\mathbf{m}_j^T = (\mathbf{H_d}\ \mathbf{e})^T \mathbf{H_d} = \mathbf{e}^T \mathbf{G_d}\ . \tag{G.1}$$

To see this, first observe that $\mathbf{H_d}\ \mathbf{e}$ has ones in the positions where the corresponding rows are not orthogonal to $\mathbf{e}_j$. Then, $(\mathbf{H_d}\ \mathbf{e})^T \mathbf{H_d}$ selects and sums up the rows in $\mathbf{H}_{\mathbf{d},\mathbf{e}_j}$.

According to Eq. (G.1), the rows of $\mathbf{M}$ are linear combinations of rows of $\mathbf{G_d}$ and thus are in the row space of $\mathbf{G_d}$. □

For completeness, we also give the success probability that the real secret $\mathbf{s}$ lies in $\ker(\mathbf{M})$.

Proposition G.2 (Theorem 3.1 in Ref. [30] restated)**.** *Given* $(\mathbf{H}, \mathbf{s}) \in \mathcal{H}^{\mathrm{QRC}}_{n,m,q}$, *randomly sample a vector* $\mathbf{d} \in \{0,1\}^n$ *and let* $\mathbf{M}$ *be the binary matrix obtained from Meta-Algorithm 3. If* $\mathbf{G_s d} = 0$, *then we have* $\mathbf{Ms} = \mathbf{0}$, *which happens with probability* $1/2$ *over all choices of* $\mathbf{d}$.

Proof. First, note that for the i-th row of $\mathbf{M}$,

$$\mathbf{m}_i \cdot \mathbf{s} = \sum_{\substack{\mathbf{p}^T \in \mathrm{row}(\mathbf{H}) \\ \mathbf{p}\cdot\mathbf{d} = \mathbf{p}\cdot\mathbf{e}_i = 1}} \mathbf{p} \cdot \mathbf{s} = \sum_{\mathbf{p} \in \mathrm{row}(\mathbf{H})} (\mathbf{p}\cdot\mathbf{s})(\mathbf{p}\cdot\mathbf{d})(\mathbf{p}\cdot\mathbf{e}_i)\ , \tag{G.2}$$

since each term equals 1 if and only if it has inner product 1 with $\mathbf{d}$, $\mathbf{e}$ and $\mathbf{s}$ simultaneously. The above transformation is to take the conditions in the summation up to the summand, and we can take the term $\mathbf{p}\cdot\mathbf{s} = 1$ down to the summation. That is, we can write

$$\mathbf{m}_i \cdot \mathbf{s} = \sum_{\mathbf{p} \in \mathrm{row}(\mathbf{H_s})} (\mathbf{p}\cdot\mathbf{d})(\mathbf{p}\cdot\mathbf{e}_i)\ , \tag{G.3}$$

which can be seen to be a quantity only depending on $\mathbf{H_s}$. Further observe that the above is the inner product between $\mathbf{H_s d}$ and $\mathbf{H_s e}_i$, i.e.,

$$\mathbf{m}_i \cdot \mathbf{s} = (\mathbf{H_s e}_i) \cdot (\mathbf{H_s d}) = \mathbf{e}_i^T \mathbf{G_s d}\ . \tag{G.4}$$

Therefore, if $\mathbf{G_s d} = \mathbf{0}$, then $\mathbf{m}_i \cdot \mathbf{s} = 0$ for every i, which means $\mathbf{Ms} = \mathbf{0}$. That is, the verifier's secret lies in the kernel of $\mathbf{M}$ if $\mathbf{d}$ lies in the kernel of $\mathbf{G_s}$. If $\mathbf{H_s}$ generates a QRC, then $\mathrm{rank}(\mathbf{G_s}) = 1$. Then, the probability that $\mathbf{d}$ lies in the kernel of $\mathbf{G_s}$ is $2^{n-1}/2^n = 1/2$. □

G.2 Classical sampling

Here, we give two classical sampling algorithms that given an IQP circuit C and a candidate set $S = \{\mathbf{s}_1, \cdots, \mathbf{s}_t\}$ with $t \le n$ as input, output samples that have the correct correlation function on all candidate secrets in the set. We first consider a simple case here, where the Gram matrix $\mathbf{G}_{\mathbf{s}_i} = \mathbf{H}_{\mathbf{s}_i}^T \mathbf{H}_{\mathbf{s}_i}$ associated with each candidate secret $\mathbf{s}_i$ has the same rank. Then, if the samples are from a quantum computer, the probability bias relative to every candidate secret should be the

Parameter: number of samples T.

procedure CLASSICALSAMPLING($\mathbf{S}, \beta$)
    Solve $\mathbf{Sy} = \mathbf{1}$ for a specific solution $\mathbf{y}'$.
    Find the basis $\{\mathbf{y}_1, \cdots, \mathbf{y}_k\}$ of $\ker(\mathbf{S})$ ▷ k is the dimension of $\ker(\mathbf{S})$.
    for $j = 1, 2, \cdots, T$ **do**
        Randomly sample $(c_1, \cdots, c_k) \in \mathbb{F}^k$.
        With probability β, set $\mathbf{x}_j \leftarrow \sum_{i=1}^{k} c_i \mathbf{y}_i$.
        With probability $1 - \beta$, set $\mathbf{x}_j \leftarrow \mathbf{y}' + \sum_{i=1}^{k} c_i \mathbf{y}_i$.
    end for
    return $\mathbf{x}_1, \cdots, \mathbf{x}_T$
end procedure

Algorithm 3: The CLASSICALSAMPLING subroutine for the candidate set where all secrets are associated with the same bias.

same, denoted as β. Given this candidate set, a classical prover can use Algorithm 3 to mimic the quantum behavior. Here, the matrix $\mathbf{S}$ is defined to be a $t \times n$ matrix whose i-th row is $\mathbf{s}_i^T$. The output bit strings will have probability β to be orthogonal to all vectors in the candidate set S, and probability $1 - \beta$ to have inner product 1 with them. Therefore, the generated samples will have correct bias with every vector in the candidate set and hence the correct correlation function. The condition for Algorithm 3 to work is that the all-ones vector $\mathbf{1}$ needs to be in the column space of $\mathbf{S}$. Otherwise, the specific solution $\mathbf{y}'$ cannot be found. A sufficient condition is that the candidate vectors are linearly independent. Then, the matrix $\mathbf{S}$ will have full row rank, and the all-ones vector $\mathbf{1}$ will be in the column space of $\mathbf{S}$.

Next, we do not require the associated biases $\beta_1, \cdots, \beta_t$ to be the same. We present a similar sampling algorithm to Algorithm 3 to output samples that mimic what a quantum computer will output. For the sake of illustration, we assume that the associated biases are all different, denoted as $\{\beta_1, \beta_2, \cdots, \beta_t\}$, but the following discussion can be easily generalized to the case where some β_i's are the same. As before, the attacker does not have extra information to judge which one is the correct secret, even though the correct secret is in the candidate set. So he would have to generate samples that have bias β_1 with $\mathbf{s}_1$, β_2 with $\mathbf{s}_2$, and so on. Below, the algorithm for generating such samples is shown in Algorithm 4. Again, we transform the set S into a $t \times n$ matrix $\mathbf{S}$.

The correctness of the sampling algorithm can be easily seen via a sanity check. But one problem is whether the linear system $\mathbf{Sy} = \mathbf{b}_j$ has solutions or not. If nonzero solutions can be found for every linear systems, then $\mathbf{b}_j$'s are all in the column space of $\mathbf{S}$, which implies that the rank of $\mathbf{S}$ is t. Since there are t rows in $\mathbf{S}$, a necessary and sufficient condition for the sampling algorithm to work is that $\{\mathbf{s}_1, \cdots, \mathbf{s}_t\}$ are linearly independent. This condition can be relaxed if some of the β_j's are the same.

G.3 The probability of choosing a good d

Here, we prove Proposition 5.4.

Proof. First, $\mathbf{G_s d} = \mathbf{H_s}^T \mathbf{H_s d}$ and $\mathbf{H_s d}$ is a vector, where the positions of ones gives the indices of

Input: a binary matrix $\mathbf{S} \in \mathbb{F}^{t\times n}$; biases $\beta_1 > \beta_2 > \cdots > \beta_t$.
Parameter: number of samples T.
Output: $\mathbf{x}_1, \cdots, \mathbf{x}_T \in \mathbb{F}^n$.

```
Find the basis of ker(S), denoted as {y_1, ..., y_k}.
for j = 1, 2, ..., t do
    Define b_j to be a binary vector whose last j entries are all zero.
    Solve a specific solution y'_j for Sy = b_j.
end for
for i = 1, 2, ..., T do
    Randomly sample (c_1, ..., c_k) ∈ F^k.
    With probability β_t, set x_i ← Σ_{i=1}^k c_i y_i.
    With probability β_{t-1} − β_t, set x_i ← y'_t + Σ_{i=1}^k c_i y_i.
    With probability β_{t-2} − β_{t-1}, set x_i ← y'_{t-1} + Σ_{i=1}^k c_i y_i.
    ⋮
    With probability β_1 − β_2, set x_i ← y'_2 + Σ_{i=1}^k c_i y_i.
    With probability 1 − β_1, set x_i ← y'_1 + Σ_{i=1}^k c_i y_i.
end for
return x_1, ..., x_T
```

Algorithm 4: The CLASSICALSAMPLING subroutine for the candidate set where all vectors are associated with different biases.

the rows in $\mathbf{H_s}$ that have inner product 1 with $\mathbf{d}$. Therefore, the ones of $\mathbf{H_s d}$ correspond to the rows in $\mathbf{H}$ that have inner product 1 with both $\mathbf{s}$ and $\mathbf{d}$. Moreover, if the vector $\mathbf{H_s d}$ is multiplied to $\mathbf{H_s}^T$ on the right, then those rows are summed up, i.e.,

$$\mathbf{G_s d} = \mathbf{H_s}^T \mathbf{H_s d} = \sum_{\substack{\mathbf{p}^T \in \mathrm{row}(\mathbf{H}) \\ \mathbf{p}\cdot\mathbf{d} = \mathbf{p}\cdot\mathbf{s} = 1}} \mathbf{p} \,. \tag{G.5}$$

Similarly, we have

$$\mathbf{G_d s} = \mathbf{H_d}^T \mathbf{H_d s} = \sum_{\substack{\mathbf{p}^T \in \mathrm{row}(\mathbf{H}) \\ \mathbf{p}\cdot\mathbf{d} = \mathbf{p}\cdot\mathbf{s} = 1}} \mathbf{p} \,. \tag{G.6}$$

Thus, $\mathbf{G_s d} = \mathbf{G_d s}$. If we want $\mathbf{s}$ to lie in $\ker(\mathbf{G_d})$, then $\mathbf{d}$ needs to lie in $\ker(\mathbf{G_s})$, which happens with probability

$$\frac{2^{n-g}}{2^n} = 2^{-g} \,, \tag{G.7}$$

for a random $\mathbf{d}$. □

G.4 Size of $\ker(\mathbf{G_d})$

Here, we prove Theorem 5.5.

Proof. First, observe that the rows in $\mathbf{G_d}$ are formed by linear combination of rows in $\mathbf{H_d}$, which means the rows space of $\mathbf{G_d}$ is no larger than that of $\mathbf{H_d}$, and $\mathrm{rank}(\mathbf{G_d}) \leq \mathrm{rank}(\mathbf{H_d})$. So, the dimension of $\ker(\mathbf{G_d})$ is

$$n - \mathrm{rank}(\mathbf{G_d}) \geq n - \mathrm{rank}(\mathbf{H_d}) \,. \tag{G.8}$$

In expectation, the number of rows $r(\mathbf{H_d})$ in $\mathbf{H_d}$ is

$$\mathbb{E}_{\mathbf{d}}[r(\mathbf{H_d})] = \mathbb{E}_{\mathbf{d}}\left[\sum_{\mathbf{p}^T \in \mathrm{row}(\mathbf{H})} \mathbf{p} \cdot \mathbf{d}\right] = \sum_{\mathbf{p}^T \in \mathrm{row}(\mathbf{H})} \mathbb{E}_{\mathbf{d}}\left[\mathbf{p} \cdot \mathbf{d}\right] = \frac{m}{2} \,, \tag{G.9}$$

since $\mathbb{E}_{\mathbf{d}}\left[\mathbf{p} \cdot \mathbf{d}\right] = 1/2$ for every row $\mathbf{p}^T$. Since $\mathrm{rank}(\mathbf{H_d}) \leq r(\mathbf{H_d})$, we have $\dim(\ker(\mathbf{G_d})) \geq n - m/2$ in expectation. □