8-23-2010

# Convolutional entanglement distillation

Mark M. Wilde
*Université McGill*

Hari Krovi
*Université McGill*

Todd A. Brun
*University of Southern California*



Recommended Citation
Wilde, M., Krovi, H., & Brun, T. (2010). Convolutional entanglement distillation. *IEEE International Symposium on Information Theory - Proceedings*, 2657-2661. https://doi.org/10.1109/ISIT.2010.5513666

# Convolutional Entanglement Distillation

Mark M. Wilde, Hari Krovi, and Todd A. Brun



***Abstract*—We develop a theory of entanglement distillation that exploits a convolutional coding structure. We provide a method for converting an arbitrary classical binary or quaternary convolutional code into a convolutional entanglement distillation protocol. The imported classical convolutional code does not have to be dual-containing or self-orthogonal. The yield and error-correcting properties of such a protocol depend respectively on the rate and error-correcting properties of the imported classical convolutional code. A convolutional entanglement distillation protocol has several other benefits. Two parties sharing noisy ebits can distill noiseless ebits "online" as they acquire more noisy ebits. Distillation yield is high and decoding complexity is simple for a convolutional entanglement distillation protocol. Our theory of convolutional entanglement distillation reduces the problem of finding a good convolutional entanglement distillation protocol to the well-established problem of finding a good classical convolutional code.**

***Index Terms*—quantum convolutional codes, convolutional entanglement distillation, quantum information theory, entanglement-assisted quantum codes, catalytic codes**

## I. Introduction

The theory of quantum error correction [1], [2], [3], [4], [5] plays a prominent role in the practical realization and engineering of quantum computing and communication devices. The first quantum error-correcting codes [1], [2], [3], [5] are strikingly similar to classical block codes [6] in their operation and performance. Quantum error-correcting codes restore a noisy, decohered quantum state to a pure quantum state. A *stabilizer* [4] quantum error-correcting code appends ancilla qubits to qubits that we want to protect. A unitary encoding circuit rotates the global state into a subspace of a larger Hilbert space. This highly entangled, encoded state corrects for local noisy errors. Figure 1 illustrates the above procedures for encoding a stabilizer code. A quantum error-correcting code makes quantum computation and quantum communication practical by providing a way for a sender and receiver to simulate a noiseless qubit channel given a noisy qubit channel that has a particular error model.

The stabilizer theory of quantum error correction allows one to import some classical binary or quaternary codes for use as a quantum code. The only "catch" when importing is that the classical code must satisfy the dual-containing or self-orthogonality constraint. Researchers have found many examples of classical codes satisfying this constraint [5], but most classical codes do not.

Brun, Devetak, and Hsieh extended the standard stabilizer theory of quantum error correction by developing the entanglement-assisted stabilizer formalism [7], [8]. They included entanglement as a resource that a sender and receiver can exploit for a quantum error-correcting code. They provided a "direct-coding" construction in which a sender and receiver can use ancilla qubits and ebits[1] in a quantum code. Gottesman later showed that their construction is optimal [9]—it gives the minimum number of ebits required for the entanglement-assisted quantum code. The benefit of including shared entanglement is that one can import an arbitrary classical binary or quaternary code for use as an entanglement-assisted quantum code. Another benefit of using shared entanglement, in addition to being able to import an arbitrary classical linear code, is that the performance of the original classical code determines the performance of the resulting quantum code. The entanglement-assisted stabilizer formalism thus is a significant and powerful extension of the stabilizer formalism.

The goal of entanglement distillation resembles the goal of quantum error correction [10], [11]. An entanglement distillation protocol extracts noiseless, maximally-entangled ebits from a larger set of noisy ebits. A sender and receiver can use these noiseless ebits as a resource for several quantum communication protocols [12], [13].

Bennett et al. showed that a strong connection exists between quantum error-correcting codes and entanglement distillation and demonstrated a method for converting an arbitrary quantum error-correcting code into a one-way entanglement distillation protocol [11]. A one-way entanglement distillation protocol utilizes one-way classical communication between sender and receiver to carry out the distillation procedure. Shor and Preskill improved upon Bennett et al.'s method by avoiding the use of ancilla qubits and gave a simpler method for converting an arbitrary CSS quantum error-correcting code into an entanglement distillation protocol [14]. Nielsen and Chuang showed how to convert a stabilizer quantum error-correcting code into a stabilizer entanglement distillation protocol [15]. Luo and Devetak then incorporated shared entanglement to demonstrate how to convert an entanglement-assisted stabilizer code into an entanglement-assisted entanglement distillation protocol [16]. All of the above constructions exploit the relationship between quantum error correction and entanglement distillation—we further exploit the connection in this paper by forming a *convolutional* entanglement distillation protocol.

Several authors have recently contributed toward a theory of quantum convolutional codes [17], [18], [19], [20]. Quantum convolutional codes are useful in a communication context where a sender has a large stream of qubits to send to a receiver. Quantum convolutional codes are similar to classical convolutional codes in their operation and performance [18],

Mark M. Wilde, Hari Krovi, and Todd A. Brun are with the Communication Sciences Institute of the Ming Hsieh Department of Electrical Engineering at the University of Southern California, Los Angeles, California 90089 USA (E-mail: mark.wilde@usc.edu; krovi@usc.edu; tbrun@usc.edu).

[1] An ebit is a nonlocal bipartite Bell state $\left|\Phi^{+}\right\rangle = \left(|00\rangle + |11\rangle\right)/\sqrt{2}$.

[20]. Classical convolutional codes have some advantages over classical block codes such as superior code rates and lower decoding complexity [21]. Their quantum counterparts enjoy these same advantages over quantum block codes [20].

The development of quantum convolutional codes has been brief but successful. Chau was the first to construct some quantum convolutional codes [22], [23], though some authors [20] argue that his construction is not a true quantum convolutional code. Several authors have established a working theory of quantum convolutional coding based on the stabilizer formalism and classical self-orthogonal codes over the finite field $\mathbb{F}_4$ [17], [18], [19], [20]. Others have also provided a practical way for realizing "online" encoding and decoding circuits for quantum convolutional codes [17], [18], [24], [25]. These successes have led to a theory of quantum convolutional coding which is useful but not complete. We add to the usefulness of the quantum convolutional theory by considering entanglement distillation and shared entanglement.

In this paper, our main contribution is a theory of convolutional entanglement distillation. Our theory allows us to import the entirety of classical convolutional coding theory for use in entanglement distillation. The task of finding a good convolutional entanglement distillation protocol now becomes the well-established task of finding a good classical convolutional code.

We begin in Section III by showing how to construct a convolutional entanglement distillation protocol from an arbitrary quantum convolutional code. We translate earlier protocols [14], [15] for entanglement distillation of a block of noisy ebits to the convolutional setting. Our convolutional entanglement distillation protocol possesses several benefits—it has a higher distillation yield and lower decoding complexity than a block entanglement distillation protocol. A convolutional entanglement distillation protocol has the additional benefit of distilling entanglement "online." This online property is useful because the sender and receiver can distill entanglement "on the fly" as they obtain more noisy ebits. This translation from a quantum convolutional code to an entanglement distillation protocol is useful because it paves the way for our major contribution.

Our major advance is a method for constructing a convolutional entanglement distillation protocol when the sender and receiver initially share some noiseless ebits. All prior quantum convolutional work requires the code to satisfy the restrictive self-orthogonality constraint, and authors performed specialized searches for classical convolutional codes that meet this constraint [17], [18], [19], [20]. We lift this constraint by allowing shared noiseless entanglement. The benefit of convolutional entanglement distillation with entanglement assistance is that we can import an *arbitrary* classical binary or quaternary convolutional code for use in a convolutional entanglement distillation protocol. The error-correcting properties for the convolutional entanglement distillation protocol follow directly from the properties of the imported classical code. Thus we can apply the decades of research on classical convolutional coding theory with many of the benefits of the convolutional structure carrying over to the quantum domain.

We organize our work as follows. In Section II, we review the stabilizer theory for quantum error correction and entanglement distillation. The presentation of the mathematics is similar in style to Refs. [20], [8]. The stabilizer review includes a review of the standard stabilizer theory (Section II-A), the entanglement-assisted stabilizer theory (Section II-B), convolutional stabilizer codes (Section II-C), stabilizer entanglement distillation (Section II-D), and entanglement-assisted entanglement distillation (Section II-E). We provide a small contribution in the Appendix—a simple algorithm to determine an encoding circuit and the optimal number of ebits required for an entanglement-assisted block code. The original work [8] gave two theorems relevant to the encoding circuit, but the algorithm we present here is simpler. In Section III, we show how to convert an arbitrary quantum convolutional code into a convolutional entanglement distillation protocol. In Section IV, we provide several methods and examples for constructing convolutional entanglement distillation protocols where two parties possess a few initial noiseless ebits. These initial noiseless ebits act as a catalyst for the convolutional distillation protocol. The constructions in Section IV make it possible to import an arbitrary classical binary or quaternary convolutional code for use in convolutional entanglement distillation.

## II. Review of the Stabilizer Formalism

### A. Standard Stabilizer Formalism for Quantum Block Codes

The stabilizer formalism exploits elements of the Pauli group $\Pi$ in formulating quantum error-correcting codes. The set $\Pi = \{I, X, Y, Z\}$ consists of the Pauli operators:

$$I \equiv \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},\ X \equiv \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix},\ Y \equiv \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix},\ Z \equiv \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}.$$

The above operators act on a single qubit—a state in a two-dimensional Hilbert space. Operators in $\Pi$ have eigenvalues $\pm 1$ and either commute or anti-commute. The set $\Pi^n$ consists of $n$-fold tensor products of Pauli operators:

$$\Pi^n = \left\{ \begin{array}{c} e^{i\phi} A_1 \otimes \cdots \otimes A_n : \forall j \in \{1, \ldots, n\}, \\ A_j \in \Pi, \quad \phi \in \{0, \pi/2, \pi, 3\pi/2\} \end{array} \right\}. \quad (1)$$

Elements of $\Pi^n$ act on a quantum register of $n$ qubits. We occasionally omit tensor product symbols in what follows so that $A_1 \cdots A_n \equiv A_1 \otimes \cdots \otimes A_n$. The $n$-fold Pauli group $\Pi^n$ plays an important role for both the encoding circuit and the error-correction procedure of a quantum stabilizer code over $n$ qubits.

Let us define an $[n, k]$ stabilizer quantum error-correcting code to encode $k$ logical qubits into $n$ physical qubits. The rate of such a code is $k/n$. Its stabilizer $\mathcal{S}$ is an abelian subgroup of the $n$-fold Pauli group $\Pi^n$: $\mathcal{S} \subset \Pi^n$. $\mathcal{S}$ does not contain the operator $-I^{\otimes n}$. The simultaneous $+1$-eigenspace of the operators constitutes the *codespace*. The codespace has dimension $2^k$ so that we can encode $k$ qubits into it. The stabilizer $\mathcal{S}$ has a minimal representation in terms of $n-k$ independent generators $\{g_1, \ldots, g_{n-k} \mid \forall i \in \{1, \ldots, n-k\},\ g_i \in \mathcal{S}\}$. The generators are independent in the sense that none of them is a product of any other two (up to a global phase). The operators $g_1, \ldots, g_{n-k}$ function in the same way as a parity check matrix does for a classical linear block code. Figure 1 illustrates the operation of a stabilizer code.

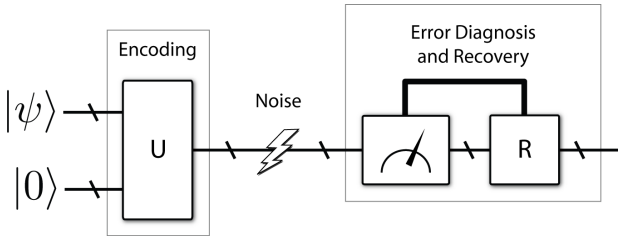


Fig. 1. The operation of a stabilizer code. Thin lines denote quantum information and thick lines denote classical information. Slanted bars denote multiple qubits. A sender encodes a multi-qubit state $|\psi\rangle$ with the help of some ancilla qubits $|0\rangle$. She sends the encoded state over a noisy quantum channel. The receiver performs multi-qubit measurements to extract information about the errors. He finally performs a recovery operation $R$ to reverse the channel error.

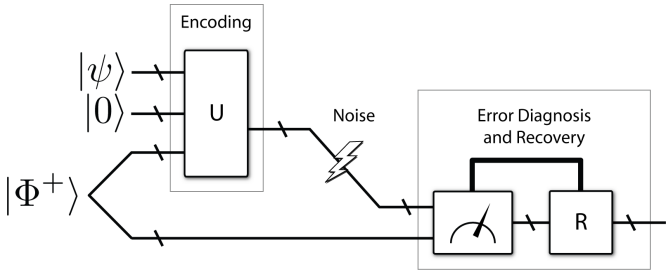


Fig. 2. The operation of an entanglement-assisted quantum error-correcting code. The sender encodes quantum information in state $|\psi\rangle$ with the help of local ancilla qubits $|0\rangle$ and her half of a set of shared ebits $|\Phi^+\rangle$. She then sends her qubits over a noisy quantum channel. The channel does not corrupt the receiver's half of the set of shared ebits. The receiver performs multi-qubit measurements on all of the qubits to diagnose the channel error. He performs a recovery unitary $R$ to reverse the channel error.

One of the fundamental notions in quantum error correction theory is that it suffices to correct a discrete error set with support in the Pauli group $\Pi^n$ [1]. Suppose that the errors affecting an encoded quantum state are a subset $\mathcal{E}$ of the Pauli group $\Pi^n$: $\mathcal{E} \subset \Pi^n$. An error $E \in \mathcal{E}$ that affects an encoded quantum state either commutes or anticommutes with any particular element $g$ in $\mathcal{S}$. The error $E$ is correctable if it anticommutes with an element $g$ in $\mathcal{S}$. An anticommuting error $E$ is detectable by measuring each element $g$ in $\mathcal{S}$ and computing a syndrome $\mathbf{r}$ identifying $E$. The syndrome is a binary vector $\mathbf{r}$ with length $n-k$ whose elements identify whether the error $E$ commutes or anticommutes with each $g \in \mathcal{S}$. An error $E$ that commutes with every element $g$ in $\mathcal{S}$ is correctable if and only if it is in $\mathcal{S}$. It corrupts the encoded state if it commutes with every element of $\mathcal{S}$ but does not lie in $\mathcal{S}$. So we compactly summarize the stabilizer error-correcting conditions: a stabilizer code can correct any errors $E_1, E_2$ in $\mathcal{E}$ if $E_1^\dagger E_2 \notin \mathcal{Z}(\mathcal{S})$ or $E_1^\dagger E_2 \in \mathcal{S}$ where $\mathcal{Z}(\mathcal{S})$ is the centralizer of $\mathcal{S}$.

### *B. Entanglement-Assisted Stabilizer Formalism*

The entanglement-assisted stabilizer formalism extends the standard stabilizer formalism by including shared entanglement [7], [8]. Figure 2 demonstrates the operation of a generic entanglement-assisted stabilizer code.

The advantage of entanglement-assisted stabilizer codes is that the sender can exploit the error-correcting properties of an arbitrary set of Pauli operators. The sender's Pauli operators do not necessarily have to form an abelian subgroup of $\Pi^n$. The sender can make clever use of her shared ebits so that the global stabilizer is abelian and thus forms a valid quantum error-correcting code.

We review the construction of an entanglement-assisted code. Suppose that there is a nonabelian subgroup $\mathcal{S} \subset \Pi^n$ of size $n-k = 2c+s$. Application of the fundamental theorem of symplectic geometry[2] [26] (Lemma 1 in [7]) states that there exists a minimal set of independent generators $\left\{\bar{Z}_1, \ldots, \bar{Z}_{s+c}, \bar{X}_{s+1}, \ldots, \bar{X}_{s+c}\right\}$ for $\mathcal{S}$ with the following commutation relations:

$$\begin{aligned}
\left[\bar{Z}_i, \bar{Z}_j\right] &= 0 \qquad \forall i, j, \\
\left[\bar{X}_i, \bar{X}_j\right] &= 0 \qquad \forall i, j, \\
\left[\bar{X}_i, \bar{Z}_j\right] &= 0 \qquad \forall i \neq j, \\
\left\{\bar{X}_i, \bar{Z}_i\right\} &= 0 \qquad \forall i.
\end{aligned} \tag{2}$$

The decomposition of $\mathcal{S}$ into the above minimal generating set determines that the code requires $s$ ancilla qubits and $c$ ebits. The code requires an ebit for every anticommuting pair in the minimal generating set. The simple reason for this requirement is that an ebit is a simultaneous $+1$-eigenstate of the operators $\{XX, ZZ\}$. The second qubit in the ebit transforms the anticommuting pair $\{X, Z\}$ into a commuting pair $\{XX, ZZ\}$. The above decomposition also minimizes the number of ebits required for the code [9]—it is an optimal decomposition.

We can partition the nonabelian group $\mathcal{S}$ into two subgroups: the isotropic subgroup $\mathcal{S}_I$ and the entanglement subgroup $\mathcal{S}_E$. The isotropic subgroup $\mathcal{S}_I$ is a commuting subgroup of $\mathcal{S}$ and thus corresponds to ancilla qubits: $\mathcal{S}_I = \left\{\bar{Z}_1, \ldots, \bar{Z}_s\right\}$. The elements of the entanglement subgroup $\mathcal{S}_E$ come in anticommuting pairs and thus correspond to ebits: $\mathcal{S}_E = \left\{\bar{Z}_{s+1}, \ldots, \bar{Z}_{s+c}, \bar{X}_{s+1}, \ldots, \bar{X}_{s+c}\right\}$.

The two subgroups $\mathcal{S}_I$ and $\mathcal{S}_E$ play a role in the error-correcting conditions for the entanglement-assisted stabilizer formalism. An entanglement-assisted code corrects errors in a set $\mathcal{E} \subset \Pi^n$ if for all $E_1, E_2 \in \mathcal{E}$, $E_1^\dagger E_2 \in \mathcal{S}_I \cup (\Pi^n - \mathcal{Z}(\langle \mathcal{S}_I, \mathcal{S}_E \rangle))$.

The operation of an entanglement-assisted code is as follows. The sender performs an encoding unitary on her unprotected qubits, ancilla qubits, and her half of the ebits. The unencoded state is a simultaneous +1-eigenstate of the following operators:

$$\left\{ \begin{array}{c} Z_1, \ldots, Z_s, \\ Z_{s+1}|Z_1, \ldots, Z_{s+c}|Z_c, \\ X_{s+1}|X_1, \ldots, X_{s+c}|X_c \end{array} \right\}. \tag{3}$$

The operators to the right of the vertical bars indicate the receiver's half of the shared ebits. The encoding unitary transforms the unencoded operators to the following encoded

[2]We loosely refer to this theorem as the fundamental theorem of symplectic geometry because of its importance in symplectic geometry and in quantum coding theory.

operators:

$$\left\{ \begin{array}{c} \bar{Z}_1, \ldots, \bar{Z}_s, \\ \bar{Z}_{s+1}|Z_1, \ldots, \bar{Z}_{s+c}|Z_c, \\ \bar{X}_{s+1}|X_1, \ldots, \bar{X}_{s+c}|X_c \end{array} \right\}. \tag{4}$$

The sender transmits all of her qubits over the noisy quantum channel. The receiver then possesses the transmitted qubits and his half of the ebits. He measures the above encoded operators to diagnose the error. The last step is to correct for the error.

We give an example of an entanglement-assisted stabilizer code in the Appendix. This example highlights the main features of the theory given above.

The Appendix also gives an original algorithm that determines the encoding circuit for the sender to perform. The algorithm determines the number of ancilla qubits and the number of ebits that the code requires.

The defined rate of an entanglement-assisted quantum error-correcting code is $(k-c)/n$ [7], [8]. The authors defined the rate in this way to compare entanglement-assisted codes with standard quantum error-correcting codes.

We mention that the rate pair $(k/n, c/n)$ more properly characterizes the rate of an entanglement-assisted code because an entanglement-assisted code is a "father" code in the sense of Ref. [27]. The first number in the pair gives the rate of noiseless qubits generated per channel use and the second number gives the rate of ebits consumed per channel use. The rate pair falls in the two-dimensional capacity region for the "father" protocol. The goal of an entanglement-assisted coding strategy is for the rate pair to approach the boundary of the capacity region as the block length becomes large.

### C. Convolutional Stabilizer Codes

The block codes reviewed above are useful in quantum computing and in quantum communications. The encoding circuit for a large block code typically has a high complexity although those for modern codes do have lower complexity.

Quantum convolutional coding theory [17], [18], [19], [20] offers a different paradigm for coding quantum information. The convolutional structure is useful for a quantum communication scenario where a sender possesses a stream of qubits to send to a receiver. The encoding circuit for a quantum convolutional code has a much lower complexity than an encoding circuit needed for a large block code. It also has a repetitive pattern so that the same physical devices or the same routines can manipulate the stream of quantum information.

Quantum convolutional stabilizer codes borrow heavily from the structure of their classical counterparts [17], [18], [19], [20]. Quantum convolutional codes are similar because some of the qubits feed back into a repeated encoding unitary and give the code a memory structure like that of a classical convolutional code. The quantum codes feature online encoding and decoding of qubits. This feature gives quantum convolutional codes both their low encoding and decoding complexity and their ability to correct a larger set of errors than a block code with similar parameters.

We first review some preliminary mathematics and follow with the definition of a quantum convolutional stabilizer code [18], [20]. We end this section with a brief discussion of encoding circuits for quantum convolutional codes.

A quantum convolutional stabilizer code acts on a Hilbert space $\mathcal{H}$ that is a countably infinite tensor product of two-dimensional qubit Hilbert spaces $\{\mathcal{H}_i\}_{i\in\mathbb{Z}^+}$ where

$$\mathcal{H} = \bigotimes_{i=0}^{\infty} \mathcal{H}_i. \tag{5}$$

and $\mathbb{Z}^+ \equiv \{0, 1, \ldots\}$. A sequence $\mathbf{A}$ of Pauli matrices $\{A_i\}_{i\in\mathbb{Z}^+}$, where

$$\mathbf{A} = \bigotimes_{i=0}^{\infty} A_i, \tag{6}$$

can act on states in $\mathcal{H}$. Let $\Pi^{\mathbb{Z}^+}$ denote the set of all Pauli sequences. The support $\text{supp}(\mathbf{A})$ of a Pauli sequence $\mathbf{A}$ is the set of indices of the entries in $\mathbf{A}$ that are not equal to the identity. The weight of a sequence $\mathbf{A}$ is the size $|\text{supp}(\mathbf{A})|$ of its support. The delay $\text{del}(\mathbf{A})$ of a sequence $\mathbf{A}$ is the smallest index for an entry not equal to the identity. The degree $\deg(\mathbf{A})$ of a sequence $\mathbf{A}$ is the largest index for an entry not equal to the identity. E.g., the following Pauli sequence

$$I \quad X \quad I \quad Y \quad Z \quad I \quad I \quad \cdots, \tag{7}$$

has support $\{1, 3, 4\}$, weight three, delay one, and degree four. A sequence has finite support if its weight is finite. Let $F(\Pi^{\mathbb{Z}^+})$ denote the set of Pauli sequences with finite support. The following definition for a quantum convolutional code utilizes the set $F(\Pi^{\mathbb{Z}^+})$ in its description.

*Definition 1:* A rate $k/n$-convolutional stabilizer code with $0 \leq k \leq n$ is a commuting set $\mathcal{G}$ of all $n$-qubit shifts of a basic generator set $\mathcal{G}_0$. The basic generator set $\mathcal{G}_0$ has $n-k$ Pauli sequences of finite support:

$$\mathcal{G}_0 = \left\{ \mathbf{G}_i \in F(\Pi^{\mathbb{Z}^+}) : 1 \leq i \leq n-k \right\}. \tag{8}$$

The constraint length $\nu$ of the code is the maximum degree of the generators in $\mathcal{G}_0$. A frame of the code consists of $n$ qubits.

*Remark 1:* The above definition requires that all elements of $\mathcal{G}$ commute. In Section IV, we lift the restrictive commutative condition when we construct a convolutional entanglement distillation protocol with entanglement assistance.

A quantum convolutional code admits an equivalent definition in terms of the delay transform or $D$-transform. The $D$-transform captures shifts of the basic generator set $\mathcal{G}_0$. Let us define the $n$-qubit delay operator $D$ acting on any Pauli sequence $\mathbf{A} \in \Pi^{\mathbb{Z}^+}$ as follows:

$$D(\mathbf{A}) = I^{\otimes n} \otimes \mathbf{A}. \tag{9}$$

We can write $j$ repeated applications of $D$ as a power of $D$:

$$D^j(\mathbf{A}) = I^{\otimes jn} \otimes \mathbf{A}. \tag{10}$$

Let $D^j(\mathcal{G}_0)$ be the set of shifts of elements of $\mathcal{G}_0$ by $j$. Then the full stabilizer $\mathcal{G}$ for the convolutional stabilizer code is

$$\mathcal{G} = \bigcup_{j\in\mathbb{Z}^+} D^j(\mathcal{G}_0). \tag{11}$$

Figure 3 outlines the operation of a convolutional stabilizer code. The protocol begins with the sender encoding a stream

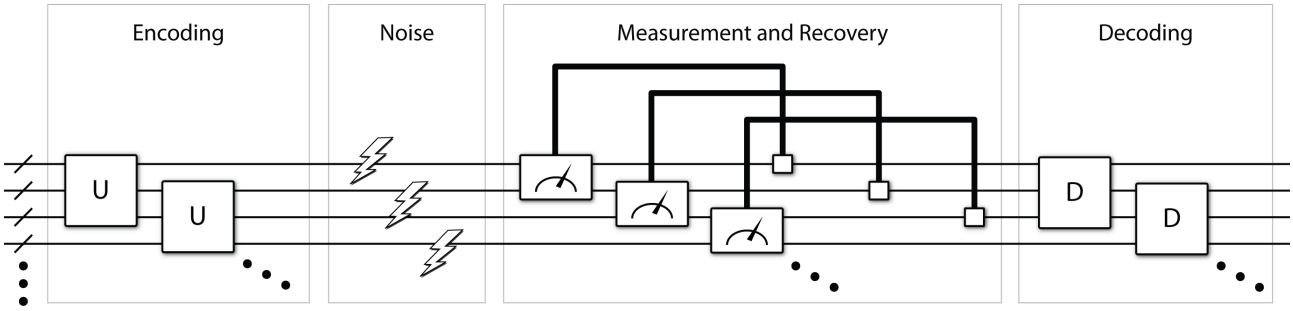


Fig. 3. An example of a quantum convolutional code. The sender applies the same unitary successively to a stream of information qubits and ancilla qubits. The convolutional structure implies that the unitary overlaps some of the same qubits. The sender transmits her qubits as soon as the unitary finishes processing them. The noisy quantum channel corrupts the transmitted qubits. The receiver performs overlapping multi-qubit measurements to diagnose channel errors and corrects for them. The receiver performs an online decoding circuit to recover the sender's original stream of information qubits.

of qubits with an online encoding circuit such as that given in [25]. The encoding circuit is "online" if it acts on a few blocks of qubits at a time. The sender transmits a set of qubits as soon as the first unitary finishes processing them. The receiver measures all the generators in $\mathcal{G}$ and corrects for errors as he receives the online encoded qubits. He finally decodes the encoded qubits with a decoding circuit. The qubits decoded from this convolutional procedure should be error free and ready for quantum computation at the receiving end.

A *finite-depth* circuit maps a Pauli sequence with finite weight to one with finite weight [18]. It does not map a Pauli sequence with finite weight to one with infinite weight. This property is important because we do not want the decoding circuit to propagate uncorrected errors into the information qubit stream [21]. A finite-depth decoding circuit corresponding to the stabilizer $\mathcal{G}$ exists by the algorithm given in [25].

*Example 1:* Forney et al. provided an example of a rate-1/3 quantum convolutional code by importing a particular classical quaternary convolutional code [19], [20]. Grassl and Rötteler determined a noncatastrophic encoding circuit for Forney et al.'s rate-1/3 quantum convolutional code [25]. The basic stabilizer and its first shift are as follows:

$$\cdots \left|\begin{array}{c} III \\ III \\ III \\ III \end{array}\right| \left|\begin{array}{c} XXX \\ ZZZ \\ III \\ III \end{array}\right| \left|\begin{array}{c} XZY \\ ZYX \\ XXX \\ ZZZ \end{array}\right| \left|\begin{array}{c} III \\ III \\ XZY \\ ZYX \end{array}\right| \left|\begin{array}{c} III \\ III \\ III \\ III \end{array}\right| \cdots \quad (12)$$

The code consists of all three-qubit shifts of the above generators. The vertical bars are a visual aid to illustrate the three-qubit shifts of the basic generators. The code can correct for an arbitrary single-qubit error in every other frame.

### *D. Stabilizer Entanglement Distillation without Entanglement Assistance*

The purpose of an $[n, k]$ entanglement distillation protocol is to distill $k$ pure ebits from $n$ noisy ebits where $0 \leq k \leq n$ [10], [11]. The yield of such a protocol is $k/n$. Two parties can then use the noiseless ebits for quantum communication protocols. Figure 4 illustrates the operation of a block entanglement distillation protocol.

The two parties establish a set of shared noisy ebits in the following way. The sender Alice first prepares $n$ Bell states

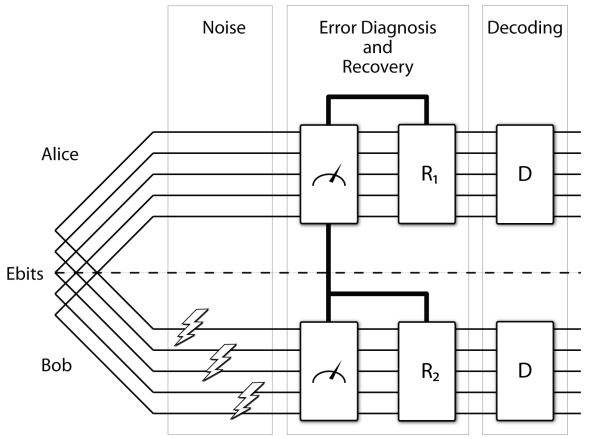


Fig. 4. An example of a block entanglement distillation protocol. A sender creates a set of noisy ebits by sending half of a set of Bell states through a noisy quantum channel. Both sender and receiver perform multi-qubit measurements to diagnose channel error. The sender transmits her measurement results to the receiver over a classical communications channel. Both perform recovery and decoding operations to obtain a set of noiseless ebits.

$|\Phi^+\rangle^{\otimes n}$ locally. She sends the second qubit of each pair over a noisy quantum channel to a receiver Bob. Let $|\Phi_n^+\rangle$ be the state $|\Phi^+\rangle^{\otimes n}$ rearranged so that all of Alice's qubits are on the left and all of Bob's qubits are on the right. The noisy channel applies a Pauli error in the error set $\mathcal{E} \subset \Pi^n$ to the set of $n$ qubits sent over the channel. The sender and receiver then share a set of $n$ noisy ebits of the form $(\mathbf{I} \otimes \mathbf{A})\,|\Phi_n^+\rangle$ where the identity $\mathbf{I}$ acts on Alice's qubits and $\mathbf{A}$ is some Pauli operator in $\mathcal{E}$ acting on Bob's qubits.

A one-way stabilizer entanglement distillation protocol uses a stabilizer code for the distillation procedure. Figure 4 highlights the main features of a stabilizer entanglement distillation protocol. Suppose the stabilizer $\mathcal{S}$ for an $[n, k]$ quantum error-correcting code has generators $g_1, \ldots, g_{n-k}$. The distillation procedure begins with Alice measuring the $n-k$ generators in $\mathcal{S}$. Let $\{\mathbf{P}_i\}$ be the set of the $2^{n-k}$ projectors that project onto the $2^{n-k}$ orthogonal subspaces corresponding to the generators in $\mathcal{S}$. The measurement projects $|\Phi_n^+\rangle$ randomly onto one of the $i$ subspaces. Each $\mathbf{P}_i$ commutes with the noisy operator

$\mathbf{A}$ on Bob's side so that

$$(\mathbf{P}_i \otimes \mathbf{I})(\mathbf{I} \otimes \mathbf{A})\left|\Phi_n^+\right\rangle = (\mathbf{I} \otimes \mathbf{A})(\mathbf{P}_i \otimes \mathbf{I})\left|\Phi_n^+\right\rangle. \quad (13)$$

The following important "Bell-state matrix identity" holds for an arbitrary matrix $\mathbf{M}$:

$$(\mathbf{M} \otimes \mathbf{I})\left|\Phi_n^+\right\rangle = \left(\mathbf{I} \otimes \mathbf{M}^T\right)\left|\Phi_n^+\right\rangle. \quad (14)$$

Then (13) is equal to the following:

$$\begin{aligned}(\mathbf{I} \otimes \mathbf{A})(\mathbf{P}_i \otimes \mathbf{I})\left|\Phi_n^+\right\rangle &= (\mathbf{I} \otimes \mathbf{A})\left(\mathbf{P}_i^2 \otimes \mathbf{I}\right)\left|\Phi_n^+\right\rangle \\ &= (\mathbf{I} \otimes \mathbf{A})\left(\mathbf{P}_i \otimes \mathbf{P}_i^T\right)\left|\Phi_n^+\right\rangle. \quad (15)\end{aligned}$$

Therefore each of Alice's projectors $\mathbf{P}_i$ projects Bob's qubits onto a subspace $\mathbf{P}_i^T$ corresponding to Alice's projected subspace $\mathbf{P}_i$. Alice restores her qubits to the simultaneous +1-eigenspace of the generators in $\mathcal{S}$. She sends her measurement results to Bob. Bob measures the generators in $\mathcal{S}$. Bob combines his measurements with Alice's to determine a syndrome for the error. He performs a recovery operation on his qubits to reverse the error. He restores his qubits to the simultaneous +1-eigenspace of the generators in $\mathcal{S}$. Alice and Bob both perform the decoding unitary corresponding to stabilizer $\mathcal{S}$ to convert their $k$ logical ebits to $k$ physical ebits.

### *E. Stabilizer Entanglement Distillation with Entanglement Assistance*

Luo and Devetak provided a straightforward extension of the above protocol [16]. Their method converts an entanglement-assisted stabilizer code into an entanglement-assisted entanglement distillation protocol.

Luo and Devetak form an entanglement distillation protocol that has entanglement assistance from a few noiseless ebits. The crucial assumption for an entanglement-assisted entanglement distillation protocol is that Alice and Bob possess $c$ noiseless ebits in addition to their $n$ noisy ebits. The total state of the noisy and noiseless ebits is

$$\left(\mathbf{I}^A \otimes (\mathbf{A} \otimes \mathbf{I})^B\right)\left|\Phi_{n+c}^+\right\rangle \quad (16)$$

where $\mathbf{I}^A$ is the $2^{n+c} \times 2^{n+c}$ identity matrix acting on Alice's qubits and the noisy Pauli operator $(\mathbf{A} \otimes \mathbf{I})^B$ affects Bob's first $n$ qubits only. Thus the last $c$ ebits are noiseless, and Alice and Bob have to correct for errors on the first $n$ ebits only.

The protocol proceeds exactly as outlined in the previous section. The only difference is that Alice and Bob measure the generators in an entanglement-assisted stabilizer code. Each generator spans over $n + c$ qubits where the last $c$ qubits are noiseless.

We comment on the yield of this entanglement-assisted entanglement distillation protocol. An entanglement-assisted code has $n - k$ generators that each have $n + c$ Pauli entries. These parameters imply that the entanglement distillation protocol produces $k + c$ ebits. But the protocol consumes $c$ initial noiseless ebits as a catalyst for distillation. Therefore the yield of this protocol is $k/n$.

In Section IV, we exploit this same idea of using a few noiseless ebits as a catalyst for distillation. The idea is similar in spirit to that developed in this section, but the mathematics and construction are different because we perform distillation in a convolutional manner.

## III. Convolutional Entanglement Distillation without Entanglement Assistance

We now show how to convert an arbitrary quantum convolutional code into a convolutional entanglement distillation protocol. Figure 5 illustrates an example of a yield-1/3 convolutional entanglement distillation protocol. The protocol has the same benefits as a quantum convolutional code: an online decoder with less decoding complexity than a block protocol, good error-correcting properties, and higher ebit yield than a block protocol. The protocol we develop in this section is useful for our major contribution presented in the next section.

We can think of our protocol in two ways. Our protocol applies when a sender Alice and a receiver Bob possess a countably infinite number of noisy ebits. Our protocol also applies as an online protocol when Alice and Bob begin with a finite number of noisy ebits and establish more as time passes. The countably infinite and online protocols are equivalent. We would actually implement the entanglement distillation protocol in the online manner, but we formulate the forthcoming mathematics with the countably infinite description. Each step in the protocol does not need to wait for the completion of its preceding step if Alice and Bob employ the protocol online.

The protocol begins with Alice and Bob establishing a set of noisy ebits. Alice prepares a countably infinite number of Bell states $|\Phi^+\rangle$ locally. She sends one half of each Bell state through a noisy quantum channel. Alice and Bob then possess a state $\rho^{AB}$ that is a countably infinite number of noisy ebits $\rho_i^{AB}$ where

$$\rho^{AB} = \bigotimes_{i=1}^{\infty} \rho_i^{AB}. \quad (17)$$

The state $\rho^{AB}$ is equivalent to the following ensemble

$$\left\{p_i, \left|\Phi^+\right\rangle_i^{AB}\right\}. \quad (18)$$

In the above, $p_i$ is the probability that the state is $|\Phi^+\rangle_i^{AB}$,

$$\left|\Phi^+\right\rangle_i^{AB} \equiv (\mathbf{I} \otimes \mathbf{A}_i)\left|\Phi_\infty^+\right\rangle^{AB}, \quad (19)$$

and $|\Phi_\infty^+\rangle^{AB}$ is the state $\left(|\Phi^+\rangle^{AB}\right)^{\otimes\infty}$ rearranged so that all of Alice's qubits are on the left and all of Bob's are on the right. $\mathbf{A}_i \in \Pi^{\mathbb{Z}^+}$ is a Pauli sequence of errors acting on Bob's side. These errors result from the noisy quantum channel. $\mathbf{I}$ is a sequence of identity matrices acting on Alice's side indicating that the noisy channel does not affect her qubits. Alice and Bob need to correct for a particular error set in order to distill noiseless ebits.

Alice and Bob employ the following strategy to distill noiseless ebits. Alice measures the $n-k$ generators in the basic set $\mathcal{G}_0$. The measurement operation projects the first $n(\nu + 1)$ ebits ($\nu$ is the constraint length) randomly onto one of $2^{n-k}$ orthogonal subspaces. Alice places the measurement outcomes in an $(n - k)$-dimensional classical bit vector $\mathbf{a}_0$. She restores her half of the noisy ebits to the simultaneous +1-eigenspace

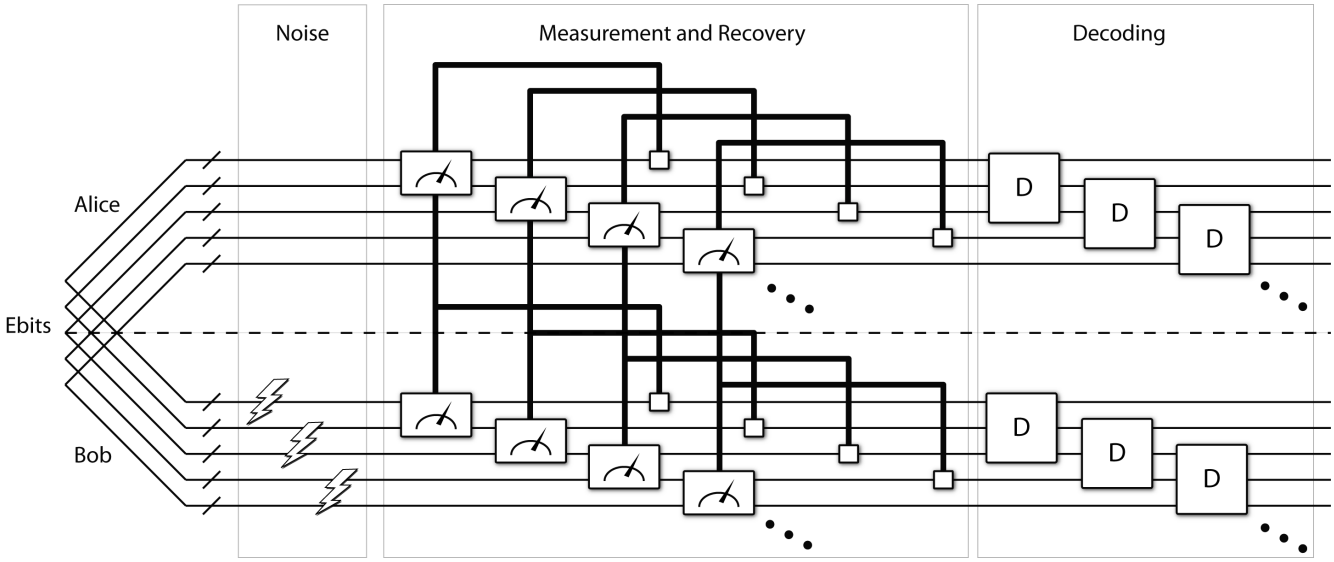


Fig. 5. An example of a convolutional entanglement distillation protocol taken from the quantum convolutional code in Ref. [20]. The code in Ref. [20] has rate 1/3 and can correct for single-qubit errors in every other frame. Alice and Bob first measure the operators in the stabilizer for the quantum convolutional code. Alice performs conditional unitaries on her qubits to restore them to the +1 eigenspace of the stabilizer code. Alice forwards her measurement results to Bob. Bob performs a maximum-likelihood decoding procedure such as Viterbi decoding [28] to determine the qubit errors. He corrects for these errors. He restores his qubits to the +1 eigenspace of the stabilizer code. Alice and Bob both perform online decoding to obtain ebits with yield 1/3.

of the generators in $\mathcal{G}_0$ if $\mathbf{a}_0$ differs from the all-zero vector. She sends $\mathbf{a}_0$ to Bob over a classical communications channel. Bob measures the generators in $\mathcal{G}_0$ and stores the measurement outcomes in a classical bit vector $\mathbf{b}_0$. Bob compares $\mathbf{b}_0$ to $\mathbf{a}_0$ by calculating an error vector $\mathbf{e}_0 = \mathbf{a}_0 \oplus \mathbf{b}_0$. He corrects for any errors that $\mathbf{e}_0$ can identify. He may have to wait to receive later error vectors before determining the full error syndrome. He restores his half of the noisy ebits to the simultaneous +1-eigenspace of the generators in $\mathcal{G}_0$ if the bit vector $\mathbf{b}_0$ indicates that his logical ebits are not in the +1-space. Alice and Bob repeat the above procedure for all shifts $D(\mathcal{G}_0)$, $D^2(\mathcal{G}_0)$, $\ldots$ of the basic generators in $\mathcal{G}_0$. Bob obtains a set $\mathcal{E}$ of classical error vectors $\mathbf{e}_i$: $\mathcal{E} = \{\mathbf{e}_i : i \in \mathbb{Z}^+\}$. Bob uses a maximum-likelihood decoding technique such as Viterbi decoding [28] or a table-lookup on the error set $\mathcal{E}$ to determine which errors occur. This error determination process is a purely classical computation. He reverses all errors after determining the syndrome.

The states that Alice and Bob possess after the above procedure are encoded logical ebits. They can extract physical ebits from these logical ebits by each performing the online decoding circuit for the code $\mathcal{G}$. The algorithm outlined in [25] gives a method for determining the online decoding circuit.

*Example 2:* We use the rate-1/3 quantum convolutional code in Example 1 to produce a yield-1/3 convolutional entanglement distillation protocol. Alice measures the generators in the stabilizer in (12) for every noisy ebit she shares with Bob. Alice communicates the result of her measurement of the first two generators to Bob. Alice restores the qubits on her side to be in the simultaneous +1-eigenspace of the first two generators. Bob measures the same first two generators. Alice measures the next two generators, communicates her results, etc. Bob compares his results to Alice's to determine the error bit vectors. Bob performs Viterbi decoding on the measurement results and corrects for errors. He rotates his states to the simultaneous +1-eigenspace of the generators. Alice and Bob perform the above procedure in an online manner according to Figure 5. Alice and Bob can decode the first six qubits after measuring the second two generators. They can decode because there is no overlap between the first two generators and any two generators after the second two generators. They use the circuit from [25] in reverse order to decode physical ebits from logical ebits. They distill ebits with yield 1/3 by using this convolutional entanglement distillation protocol. The ebit yield of 1/3 follows directly from the code rate of 1/3.

## IV. Convolutional Entanglement Distillation with Entanglement Assistance

The convolutional entanglement distillation protocol that we develop in this section operates identically to the one developed in the previous section. The measurements, classical communication, and recovery and decoding operations proceed exactly as Figure 5 indicates.

The difference between the protocol in this section and the previous one is that we now assume the sender and receiver share a few initial noiseless ebits. They use these initial ebits as a catalyst to get the protocol started. The sender and receiver require noiseless ebits for each round of the convolutional entanglement distillation protocol. They can use the noiseless ebits generated by earlier rounds for consumption in later rounds. It is possible to distill noiseless ebits in this way by catalyzing the process with a few noiseless ebits. The protocol we develop in this section is a more powerful generalization of the previous section's protocol.

The construction in this section allows sender and receiver to use an arbitrary set of Paulis for the distillation protocol.

The set does not necessarily have to be a commuting set of Paulis. The idea is similar in spirit to entanglement-assisted quantum error correction [7], [8].

The implication of the construction in this section is that we can import an arbitrary binary or quaternary classical convolutional code for use as a quantum convolutional code. We explicitly give some examples to highlight the technique for importing. The error-correcting properties and yield translate directly from the properties of the classical convolutional code. Thus the problem of finding a good convolutional entanglement distillation protocol reduces to that of finding a good classical convolutional code.

We first review some mathematics concerning the commutative properties of quantum convolutional codes. We then present our different constructions for a convolutional entanglement distillation protocol that use entanglement assistance.

### A. Commutative Properties of Quantum Convolutional Codes

We consider the commutative properties of quantum convolutional codes. We develop some mathematics that leads to the important "shifted symplectic product." The shifted symplectic product reveals the commutation relations of an arbitrary number of shifts of a set of Pauli sequences. All of our constructions following this preliminary section exploit the properties of the shifted symplectic product.

We first define the phase-free Pauli group $\left[\Pi^{\mathbb{Z}}\right]$ on a sequence of qubits. Recall that the delay transform $D$ in (9) shifts a Pauli sequence to the right by $n$. Let us assume for now that $n = 1$. Let $\Pi^{\mathbb{Z}}$ denote the set of all countably infinite Pauli sequences. The set $\Pi^{\mathbb{Z}}$ is equivalent to the set of all one-qubit shifts of arbitrary Pauli operators:

$$\Pi^{\mathbb{Z}} = \left\{ \prod_{i\in\mathbb{Z}} D^i \left(A_i\right) : A_i \in \Pi \right\}. \tag{20}$$

We remark that $D^i\left(A_i\right) = D^i\left(A_i \otimes I^{\otimes\infty}\right)$. We make this same abuse of notation in what follows. We can define the equivalence class $\left[\Pi^{\mathbb{Z}}\right]$ of phase-free Pauli sequences:

$$\left[\Pi^{\mathbb{Z}}\right] = \left\{\beta \mathbf{A} \mid \mathbf{A} \in \Pi^{\mathbb{Z}}, \beta \in \mathbb{C}, |\beta| = 1\right\}. \tag{21}$$

We develop a relation between binary polynomials and Pauli sequences that is useful for representing the shifting nature of quantum convolutional codes. Suppose $z\left(D\right)$ and $x\left(D\right)$ are arbitrary finite-degree and finite-delay polynomials in $D$ over $\mathbb{Z}_2$

$$z\left(D\right) = \sum_i z_i D^i, \qquad z_i \in \mathbb{Z}_2 \quad \forall i \in \mathbb{Z}, \tag{22}$$

$$x\left(D\right) = \sum_i x_i D^i, \qquad x_i \in \mathbb{Z}_2 \quad \forall i \in \mathbb{Z}, \tag{23}$$

where $\mathrm{del}(z\left(D\right))$, $\mathrm{del}(x\left(D\right))$, $\deg(z\left(D\right))$, $\deg(x\left(D\right)) < \infty$. Suppose

$$u\left(D\right) = \left(z\left(D\right), x\left(D\right)\right) \in \left(\mathbb{Z}_2\left(D\right)\right)^2, \tag{24}$$

where $\left(\mathbb{Z}_2\left(D\right)\right)^2$ indicates the direct product $\mathbb{Z}_2\left(D\right) \times \mathbb{Z}_2\left(D\right)$. Let us employ the following shorthand:

$$u\left(D\right) = \left(z\left(D\right) | x\left(D\right)\right). \tag{25}$$

Let $N$ be a map from the binary polynomials to the Pauli sequences, $N : \left(\mathbb{Z}_2\left(D\right)\right)^2 \to \Pi^{\mathbb{Z}}$, where

$$N\left(u\left(D\right)\right) = \prod_i D^i\left(Z^{z_i} X^{x_i}\right). \tag{26}$$

Let $v\left(D\right) = \left(z'\left(D\right) | x'\left(D\right)\right)$ where $v\left(D\right) \in \left(\mathbb{Z}_2\left(D\right)\right)^2$. The map $N$ induces an isomorphism

$$[N] : \left(\mathbb{Z}_2\left(D\right)\right)^2 \to \left[\Pi^{\mathbb{Z}}\right], \tag{27}$$

because addition of binary polynomials is equivalent to multiplication of Pauli elements up to a global phase:

$$\left[N\left(u\left(D\right) + v\left(D\right)\right)\right] = \left[N\left(u\left(D\right)\right)\right]\left[N\left(v\left(D\right)\right)\right]. \tag{28}$$

The above isomorphism is a powerful way to capture the infiniteness and shifting nature of convolutional codes with finite-degree and finite-delay polynomials over the binary field $\mathbb{Z}_2$.

Recall from Definition 1 that a commuting set comprising a basic set of Paulis and all their shifts specifies a quantum convolutional code. How can we capture the commutation relations of a Pauli sequence and all of its shifts? The *shifted* symplectic product $\odot$, where

$$\odot : \left(\mathbb{Z}_2\left(D\right)\right)^2 \times \left(\mathbb{Z}_2\left(D\right)\right)^2 \to \mathbb{Z}_2\left(D\right), \tag{29}$$

is an elegant way to do so. The shifted symplectic product maps two vectors $u\left(D\right)$ and $v\left(D\right)$ to a binary polynomial with finite delay and finite degree:

$$\left(u \odot v\right)\left(D\right) = z\left(D^{-1}\right) x'\left(D\right) - x\left(D^{-1}\right) z'\left(D\right). \tag{30}$$

The symplectic orthogonality condition originally given in Ref. [18] inspires the definition for the shifted symplectic product. The shifted symplectic product is not a proper symplectic product because it fails to be alternating [26]. The alternating property requires that

$$\left(u \odot v\right)\left(D\right) = -\left(v \odot u\right)\left(D\right), \tag{31}$$

but we find instead that the following holds:

$$\left(u \odot v\right)\left(D\right) = -\left(v \odot u\right)\left(D^{-1}\right). \tag{32}$$

Every vector $u\left(D\right) \in \mathbb{Z}_2\left(D\right)^2$ is self-time-reversal antisymmetric with respect to $\odot$:

$$\left(u \odot u\right)\left(D\right) = -\left(u \odot u\right)\left(D^{-1}\right) \qquad \forall u\left(D\right) \in \mathbb{Z}_2\left(D\right)^2. \tag{33}$$

Every binary vector is also self-time-reversal symmetric with respect to $\odot$ because addition and subtraction are the same over $\mathbb{Z}_2$. We employ the addition convention from now on and drop the minus signs. The shifted symplectic product is a binary polynomial in $D$. We write its coefficients as follows:

$$\left(u \odot v\right)\left(D\right) = \sum_{i\in\mathbb{Z}} \left(u \odot v\right)_i \ D^i. \tag{34}$$

The coefficient $\left(u \odot v\right)_i$ captures the commutation relations of two Pauli sequences for $i$ $n$-qubit shifts of one of the sequences:

$$N\left(u\left(D\right)\right) D^i\left(N\left(v\left(D\right)\right)\right) = \left(-1\right)^{\left(u\odot v\right)_i} D^i\left(N\left(v\left(D\right)\right)\right) N\left(u\left(D\right)\right). \tag{35}$$

Thus two Pauli sequences $N(u(D))$ and $N(v(D))$ commute for all shifts if and only if the shifted symplectic product $(u \odot v)(D)$ vanishes.

The next example highlights the main features of the shifted symplectic product and further emphasizes the relationship between Pauli commutation and orthogonality of the shifted symplectic product.

*Example 3:* Consider two sets of binary polynomials:

$$\begin{aligned} z_1(D) &= D, \quad x_1(D) = 1 + D^3, \\ z_2(D) &= 1 + D, \quad x_2(D) = D^3. \end{aligned}$$

We form vectors $u(D)$ and $v(D)$ from the above polynomials where

$$\begin{aligned} u(D) &= (z_1(D) \mid x_1(D)), \\ v(D) &= (z_2(D) \mid x_2(D)). \end{aligned} \tag{36}$$

The isomorphism $N$ maps the above polynomials to the following Pauli sequences:

$$\begin{aligned} N(u(D)) &= (\cdots |I|X|Z|I|X|I| \cdots), \\ N(v(D)) &= (\cdots |I|Z|Z|I|X|I| \cdots). \end{aligned} \tag{37}$$

The vertical bars between every Pauli in the sequence indicate that we are considering one-qubit shifts. We determine the commutation relations of the above sequences by inspection. $N(u(D))$ anticommutes with a shift of itself by one or two to the left or right and commutes with all other shifts of itself. $N(v(D))$ anticommutes with a shift of itself by two or three to the left or right and commutes with all other shifts of itself. $N(u(D))$ anticommutes with $N(v(D))$ shifted to the left by one or two, with the zero-shifted $N(v(D))$, and with $N(v(D))$ shifted to the right by two or three. The following shifted symplectic products give us the same information:

$$\begin{aligned} (u \odot u)(D) &= D^{-2} + D^{-1} + D + D^2, \\ (v \odot v)(D) &= D^{-3} + D^{-2} + D^2 + D^3, \\ (v \odot u)(D) &= D^{-2} + D^{-1} + 1 + D^2 + D^3. \end{aligned} \tag{38}$$

The nonzero coefficients indicate the commutation relations just as (35) claims.

A quantum convolutional code in general consists of generators with $n$ qubits per frame. Therefore, we consider the $n$-qubit extension of the definitions and isomorphism given above. Let the delay transform $D$ now shift a Pauli sequence to the right by an arbitrary integer $n$. Consider a $2n$-dimensional vector $\mathbf{u}(D)$ of binary polynomials where $\mathbf{u}(D) \in (\mathbb{Z}_2(D))^{2n}$. Let us write $\mathbf{u}(D)$ as follows

$$\begin{aligned} \mathbf{u}(D) &= (\mathbf{z}(D) \,|\, \mathbf{x}(D)), \\ &= \left( \begin{array}{ccc|ccc} z_1(D) & \cdots & z_n(D) & x_1(D) & \cdots & x_n(D) \end{array} \right), \end{aligned}$$

where $\mathbf{z}(D), \mathbf{x}(D) \in (\mathbb{Z}_2(D))^n$. Suppose

$$\begin{aligned} z_i(D) &= \sum_j z_{i,j} D^j, \\ x_i(D) &= \sum_j x_{i,j} D^j. \end{aligned} \tag{39}$$

Define a map $\mathbf{N} : (\mathbb{Z}_2(D))^{2n} \to \Pi^{\mathbb{Z}}$:

$$\begin{aligned} \mathbf{N}(\mathbf{u}(D)) = \prod_j & D^j \left( Z^{z_{1,j}} X^{x_{1,j}} \right) \\ & D^j \left( I \otimes Z^{z_{2,j}} X^{x_{2,j}} \right) \cdots D^j \left( I^{\otimes n-1} \otimes Z^{z_{n,j}} X^{x_{n,j}} \right). \end{aligned}$$

$\mathbf{N}$ is equivalent to the following map (up to a global phase)

$$\begin{aligned} \mathbf{N}(\mathbf{u}(D)) = & N(u_1(D)) (I \otimes N(u_2(D))) \\ & \cdots \left( I^{\otimes n-1} \otimes N(u_n(D)) \right), \end{aligned}$$

where

$$u_i(D) = (z_i(D) \,|\, x_i(D)). \tag{40}$$

Suppose

$$\mathbf{v}(D) = (\mathbf{z}'(D) \,|\, \mathbf{x}'(D)), \tag{41}$$

where $\mathbf{v}(D) \in (\mathbb{Z}_2(D))^{2n}$. The map $\mathbf{N}$ induces an isomorphism $[\mathbf{N}] : (\mathbb{Z}_2(D))^{2n} \to \left[\Pi^{\mathbb{Z}}\right]$ for the same reasons given in (28):

$$[\mathbf{N}(\mathbf{u}(D) + \mathbf{v}(D))] = [\mathbf{N}(\mathbf{u}(D))] [\mathbf{N}(\mathbf{v}(D))]. \tag{42}$$

The isomorphism $\mathbf{N}$ is again useful because it allows us to perform binary calculations instead of Pauli calculations.

We can again define a shifted symplectic product for the case of $n$-qubits per frame. Let $\odot$ denote the shifted symplectic product between vectors of binary polynomials:

$$\odot : (\mathbb{Z}_2(D))^{2n} \times (\mathbb{Z}_2(D))^{2n} \to \mathbb{Z}_2(D). \tag{43}$$

It maps vectors of binary polynomials to a finite-degree and finite-delay binary polynomial

$$(\mathbf{u} \odot \mathbf{v})(D) = \sum_{i=1}^{n} (u_i \odot v_i)(D), \tag{44}$$

where

$$\begin{aligned} u_i(D) &= (z_i(D) \,|\, x_i(D)), \\ v_i(D) &= (z_i'(D) \,|\, x_i'(D)). \end{aligned}$$

The standard inner product gives an alternative way to define the shifted symplectic product:

$$(\mathbf{u} \odot \mathbf{v})(D) = \mathbf{z}\left(D^{-1}\right) \cdot \mathbf{x}'(D) - \mathbf{x}\left(D^{-1}\right) \cdot \mathbf{z}'(D). \tag{45}$$

Every vector $\mathbf{u}(D) \in \mathbb{Z}_2(D)^{2n}$ is self-time-reversal symmetric with respect to $\odot$:

$$(\mathbf{u} \odot \mathbf{u})(D) = (\mathbf{u} \odot \mathbf{u})\left(D^{-1}\right) \quad \forall \mathbf{u}(D) \in \mathbb{Z}_2(D)^{2n}. \tag{46}$$

The shifted symplectic product for vectors of binary polynomials is a binary polynomial in $D$. We write its coefficients as follows:

$$(\mathbf{u} \odot \mathbf{v})(D) = \sum_{i \in \mathbb{Z}} (\mathbf{u} \odot \mathbf{v})_i \; D^i. \tag{47}$$

The coefficient $(\mathbf{u} \odot \mathbf{v})_i$ captures the commutation relations of two Pauli sequences for $i$ $n$-qubit shifts of one of the sequences:

$$\begin{aligned} \mathbf{N}(\mathbf{u}(D)) D^i (\mathbf{N}(\mathbf{v}(D))) = \\ (-1)^{(\mathbf{u} \odot \mathbf{v})_i} D^i (\mathbf{N}(\mathbf{v}(D))) \mathbf{N}(\mathbf{u}(D)). \end{aligned}$$

*Example 4:* We consider the case where $n = 4$. Consider the following vectors of polynomials:

$$\begin{bmatrix} \mathbf{z}(D) \\ \mathbf{x}(D) \\ \mathbf{z}'(D) \\ \mathbf{x}'(D) \end{bmatrix} = \begin{bmatrix} 1+D & D & 1 & D \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1+D & 1+D & 1 & D \end{bmatrix}. \tag{48}$$

Suppose

$$\begin{aligned} \mathbf{u}(D) &= (\mathbf{z}(D) | \mathbf{x}(D)), \\ \mathbf{v}(D) &= (\mathbf{z}'(D) | \mathbf{x}'(D)). \end{aligned} \tag{49}$$

The isomorphism $\mathbf{N}$ maps $\mathbf{u}(D)$ and $\mathbf{v}(D)$ to the following Pauli sequences:

$$\begin{aligned} \mathbf{N}(\mathbf{u}(D)) &= (\cdots | IIII | ZXZI | ZZIZ | IIII | \cdots), \\ \mathbf{N}(\mathbf{v}(D)) &= (\cdots | IIII | XYXI | XXIX | IIII | \cdots). \end{aligned} \tag{50}$$

We can determine the commutation relations by inspection of the above Pauli sequences. $\mathbf{N}(\mathbf{u}(D))$ anticommutes with itself shifted by one to the left or right, $\mathbf{N}(\mathbf{v}(D))$ anticommutes with itself shifted by one to the left or right, and $\mathbf{N}(\mathbf{u}(D))$ anticommutes with $\mathbf{N}(\mathbf{v}(D))$ shifted by one to the left. The following shifted symplectic products confirm the above commutation relations:

$$\begin{aligned} (\mathbf{u} \odot \mathbf{u})(D) &= D^{-1} + D, \\ (\mathbf{v} \odot \mathbf{v})(D) &= D^{-1} + D, \\ (\mathbf{u} \odot \mathbf{v})(D) &= D. \end{aligned} \tag{51}$$

We note two useful properties of the shifted symplectic product $\odot$. Suppose $f(D) \in \mathbb{Z}_2(D)$ with $\deg(f) \geq 0$. Let us denote scalar polynomial multiplication as follows:

$$(f\ \mathbf{u})(D) = \begin{bmatrix} f(D) u_1(D) & \cdots & f(D) u_n(D) \end{bmatrix}. \tag{52}$$

The following identities hold.

$$((f\ \mathbf{u}) \odot \mathbf{v})(D) = f\left(D^{-1}\right)(\mathbf{u} \odot \mathbf{v})(D), \tag{53}$$

$$(\mathbf{u} \odot (f\ \mathbf{v}))(D) = f(D)(\mathbf{u} \odot \mathbf{v})(D). \tag{54}$$

We also remark that

$$(\mathbf{u} \odot \mathbf{v})(D) = (\mathbf{v} \odot \mathbf{u})(D),$$

iff

$$(\mathbf{u} \odot \mathbf{v})(D) = (\mathbf{u} \odot \mathbf{v})\left(D^{-1}\right).$$

We exploit both of the above properties in the constructions that follow.

### *B. Yield (n-1)/n Convolutional Entanglement Distillation with Entanglement Assistance*

We present our first method for constructing a convolutional entanglement distillation protocol that uses entanglement assistance. The shifted symplectic product is a crucial component of our formulation.

Suppose Alice and Bob use one generator $\mathbf{N}(\mathbf{u}(D))$ for an entanglement distillation protocol where

$$\begin{aligned} \mathbf{u}(D) &= (\mathbf{z}(D) | \mathbf{x}(D)) \\ &= \left(\ z_1(D) \quad \cdots \quad z_n(D) \ \middle|\ x_1(D) \quad \cdots \quad x_n(D)\ \right). \end{aligned}$$

We do not impose a commuting constraint on generator $\mathbf{N}(\mathbf{u}(D))$. Alice and Bob choose generator $\mathbf{N}(\mathbf{u}(D))$ solely for its error-correcting capability.

The shifted symplectic product helps to produce a commuting generator from a noncommuting one. The shifted symplectic product of $\mathbf{u}(D)$ is

$$(\mathbf{u} \odot \mathbf{u})(D) = \sum_{i \in \mathbb{Z}} (\mathbf{u} \odot \mathbf{u})_i\ D^i. \tag{55}$$

The coefficient $(\mathbf{u} \odot \mathbf{u})_0$ for zero shifts is equal to zero because every tensor product of Pauli operators commutes with itself:

$$(\mathbf{u} \odot \mathbf{u})_0 = 0. \tag{56}$$

Recall that $\mathbf{u}(D)$ is self-time-reversal symmetric (46). We adopt the following notation for a polynomial that includes the positive-index or negative-index coefficients of the shifted symplectic product $(\mathbf{u} \odot \mathbf{u})(D)$:

$$(\mathbf{u} \odot \mathbf{u})(D)^+ = \sum_{i \in \mathbb{Z}^+} (\mathbf{u} \odot \mathbf{u})_i\ D^i, \tag{57}$$

$$(\mathbf{u} \odot \mathbf{u})(D)^- = \sum_{i \in \mathbb{Z}^-} (\mathbf{u} \odot \mathbf{u})_i\ D^i. \tag{58}$$

The following identity holds:

$$(\mathbf{u} \odot \mathbf{u})(D)^+ = (\mathbf{u} \odot \mathbf{u})\left(D^{-1}\right)^-. \tag{59}$$

Consider the following vector of polynomials:

$$\mathbf{a}(D) = \left(\ (\mathbf{u} \odot \mathbf{u})(D)^+ \ \middle|\ 1\ \right). \tag{60}$$

Its relations under the shifted symplectic product are the same as $\mathbf{u}(D)$:

$$\begin{aligned} (\mathbf{a} \odot \mathbf{a})(D) &= (\mathbf{u} \odot \mathbf{u})(D)^- + (\mathbf{u} \odot \mathbf{u})(D)^+, \\ &= (\mathbf{u} \odot \mathbf{u})(D). \end{aligned} \tag{61}$$

The vector $\mathbf{a}(D)$ provides a straightforward way to make $\mathbf{N}(\mathbf{u}(D))$ commute with all of its shifts. We augment $\mathbf{u}(D)$ with $\mathbf{a}(D)$. The augmented generator $\mathbf{u}'(D)$ is as follows:

$$\mathbf{u}'(D) = \left(\ \mathbf{z}(D) \quad (\mathbf{u} \odot \mathbf{u})(D)^+ \ \middle|\ \mathbf{x}(D) \quad 1\ \right). \tag{62}$$

The augmented generator $\mathbf{u}'(D)$ has vanishing symplectic product because the shifted symplectic product of $\mathbf{a}(D)$ nulls the shifted symplectic product of $\mathbf{u}(D)$:

$$(\mathbf{u}' \odot \mathbf{u}')(D) = 0. \tag{63}$$

The augmented generator $\mathbf{N}(\mathbf{u}'(D))$ commutes with itself for every shift and is therefore useful for convolutional entanglement distillation as outlined in Section III.

We can construct an entanglement distillation protocol using an augmented generator of this form. The first $n$ Pauli entries for every frame of generator $\mathbf{N}(\mathbf{u}'(D))$ correct errors. Entry $n+1$ for every frame of $\mathbf{N}(\mathbf{u}'(D))$ makes $\mathbf{N}(\mathbf{u}'(D))$ commute with every one of its shifts. The error-correcting properties of the code do not include errors on the last (extra) ebit of each frame; therefore, this ebit must be noiseless. It is necessary to catalyze the distillation procedure with $n\nu$ noiseless ebits where $n$ is the frame size and $\nu$ is the constraint length. The distillation protocol requires this particular amount

TABLE I
THE CONVOLUTIONAL ENTANGLEMENT DISTILLATION PROTOCOL FOR EXAMPLE 5 CORRECTS FOR A SINGLE-QUBIT ERROR IN EVERY FOURTH FRAME. HERE WE LIST THE SYNDROMES CORRESPONDING TO ERRORS $X_1$, $Y_1$, AND $Z_1$ ON THE FIRST QUBIT AND TO ERRORS $X_2$, $Y_2$, AND $Z_2$ ON THE SECOND QUBIT. THE SYNDROMES ARE UNIQUE SO THAT THE RECEIVER CAN IDENTIFY WHICH ERROR OCCURS.

| $X_1$ | $Z_1$ | $Y_1$ | $X_2$ | $Z_2$ | $Y_2$ |
|---|---|---|---|---|---|
| 1 | 0 | 1 | 1 | 0 | 1 |
| 0 | 0 | 0 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 | 0 | 0 |

because it does not correct errors and generate noiseless ebits until it has finished processing the first basic set of generators and $\nu - 1$ of its shifts. Later frames can use the noiseless ebits generated from previous frames. Therefore these initial noiseless ebits are negligible when calculating the yield. This construction allows us to exploit the error-correcting properties of an arbitrary set of Pauli matrices for a convolutional entanglement distillation protocol.

We discuss the yield of such a protocol in more detail. Our construction employs one generator with $n+1$ qubits per frame. The protocol generates $n$ noiseless ebits for every frame. But it also consumes a noiseless ebit for every frame. Every frame thus produces a net of $n-1$ noiseless ebits, and the yield of the protocol is $(n-1)/n$.

This yield of $(n-1)/n$ is superior to the yield of an entanglement distillation protocol taken from the quantum convolutional codes of Forney et al. [20]. Our construction should also give entanglement distillation protocols with superior error-correcting properties because we have no self-orthogonality constraint on the Paulis in the stabilizer.

It is possible to construct an online decoding circuit for the generator $\mathbf{u}'(D)$ by the methods given in [25]. A circuit satisfies the noncatastrophic property if the polynomial entries of all of the code generators have a greatest common divisor that is a power of the delay operator $D$ [25]. The online decoding circuit for this construction obeys the noncatastrophicity property because the augmented generator $\mathbf{u}'(D)$ contains 1 as one of its entries.

*Example 5:* Suppose we have the following generator

$$\mathbf{N}(\mathbf{u}(D)) = (\cdots|II|ZZ|IX|XZ|ZI|II|\cdots),$$

where

$$\mathbf{u}(D) = \left( \begin{array}{cc|cc} 1+D^3 & 1+D^2 & D^2 & D \end{array} \right).$$

The above generator corrects for an arbitrary single-qubit error in a span of eight qubits—four frames. Table I lists the unique syndromes for errors in a single frame. The generator anticommutes with a shift of itself by one or two to the left or right. The shifted symplectic product confirms these commutation relations:

$$(\mathbf{u} \odot \mathbf{u})(D) = D + D^2 + D^{-1} + D^{-2}.$$

Let us follow the prescription in (62) for augmenting generator $\mathbf{N}(\mathbf{u}(D))$. The following polynomial

$$\begin{aligned} \mathbf{a}(D) &= \left( \begin{array}{c|c} (\mathbf{u} \odot \mathbf{u})(D)^+ & 1 \end{array} \right), \\ &= \left( \begin{array}{c|c} D + D^2 & 1 \end{array} \right), \end{aligned} \tag{64}$$

has the same commutation relations as $\mathbf{u}(D)$:

$$(\mathbf{a} \odot \mathbf{a})(D) = (\mathbf{u} \odot \mathbf{u})(D). \tag{65}$$

We augment $\mathbf{u}(D)$ as follows:

$$\mathbf{u}'(D) = \left( \begin{array}{ccc|ccc} 1+D^3 & 1+D^2 & D+D^2 & D^2 & D & 1 \end{array} \right).$$

The overall generator now looks as follows in the Pauli representation:

$$\mathbf{N}(\mathbf{u}'(D)) = (\cdots|III|ZZX|IXZ|XZZ|ZII|III|\cdots).$$

The yield of a protocol using the above construction is 1/2. Figure 6 illustrates Bob's side of the protocol. It shows which of Bob's half of the ebits are noisy and noiseless, and it gives the measurements that Bob performs.

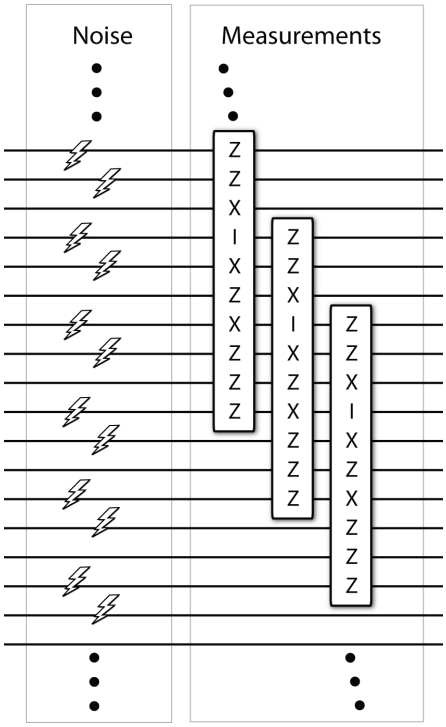


Fig. 6. The above figure illustrates Bob's side of the convolutional entanglement distillation protocol that uses entanglement assistance. The noise affects the first and second of every three ebits that Bob shares with Alice. Every third ebit that Alice and Bob share are noiseless. The measurements correspond to those in Example 5.

### C. Yield (n-m)/n Convolutional Entanglement Distillation with Entanglement Assistance

The construction in the above section uses only one generator for distillation. We generalize the above construction to a code with an arbitrary number of generators. We give an example that illustrates how to convert an arbitrary classical quaternary convolutional code into a convolutional entanglement distillation protocol.

Suppose we have the following $m$ generators

$$\{\mathbf{N}(\mathbf{u}_i(D)) : 1 \leq i \leq m\},$$

where

$$\left[\begin{array}{c} \mathbf{u}_1(D) \\ \mathbf{u}_2(D) \\ \vdots \\ \mathbf{u}_m(D) \end{array}\right] = \left[\begin{array}{c|c} \mathbf{z}_1(D) & \mathbf{x}_1(D) \\ \mathbf{z}_2(D) & \mathbf{x}_2(D) \\ \vdots & \vdots \\ \mathbf{z}_m(D) & \mathbf{x}_m(D) \end{array}\right]. \tag{66}$$

We make no assumption about the commutation relations of the above generators. We choose them solely for their error-correcting properties.

We again utilize the shifted symplectic product to design a convolutional entanglement distillation protocol with multiple generators. Let us adopt the following shorthand for the auto and cross shifted symplectic product of generators $\mathbf{u}_1(D), \ldots, \mathbf{u}_m(D)$:

$$\mathbf{u}_i^+ \equiv (\mathbf{u}_i \odot \mathbf{u}_i)(D)^+, \tag{67}$$

$$\mathbf{u}_{i,j} \equiv (\mathbf{u}_i \odot \mathbf{u}_j)(D). \tag{68}$$

Consider the following matrix:

$$\left[\begin{array}{c} \mathbf{a}_1(D) \\ \mathbf{a}_2(D) \\ \vdots \\ \mathbf{a}_m(D) \end{array}\right] = \left[\begin{array}{cccc|c} \mathbf{u}_1^+ & \mathbf{u}_{2,1} & \cdots & \mathbf{u}_{m,1} & \\ 0 & \mathbf{u}_2^+ & \cdots & \mathbf{u}_{m,2} & \\ \vdots & & \ddots & \vdots & \mathbf{I}_{m\times m} \\ 0 & \cdots & 0 & \mathbf{u}_m^+ & \end{array}\right]. \tag{69}$$

The symplectic relations of the entries $\mathbf{a}_i(D)$ are the same as the original $\mathbf{u}_i(D)$:

$$(\mathbf{a}_i \odot \mathbf{a}_j)(D) = (\mathbf{u}_i \odot \mathbf{u}_j)(D) \qquad \forall i,j \in \{1,\ldots,m\}.$$

We mention that the following matrix also has the same symplectic relations:

$$\left[\begin{array}{cccc|c} \mathbf{u}_1^+ & 0 & \cdots & 0 & \\ \mathbf{u}_{1,2} & \mathbf{u}_2^+ & \cdots & \vdots & \\ \vdots & & \ddots & 0 & \mathbf{I}_{m\times m} \\ \mathbf{u}_{1,m} & \mathbf{u}_{2,m} & \cdots & \mathbf{u}_m^+ & \end{array}\right]. \tag{70}$$

Let us rewrite (69) as follows:

$$\left[\begin{array}{c} \mathbf{a}_1(D) \\ \mathbf{a}_2(D) \\ \vdots \\ \mathbf{a}_m(D) \end{array}\right] = \left[\begin{array}{c|c} \mathbf{z}_1'(D) & \mathbf{x}_1'(D) \\ \mathbf{z}_2'(D) & \mathbf{x}_2'(D) \\ \vdots & \vdots \\ \mathbf{z}_m'(D) & \mathbf{x}_m'(D) \end{array}\right]. \tag{71}$$

The above matrix provides a straightforward way to make the original generators commute with all of their shifts. We augment the generators in (66) by the generators $\mathbf{a}_i(D)$ to get the following $m \times 2(n+m)$ matrix:

$$\mathbf{U}'(D) = \left[\begin{array}{c|c} \mathbf{Z}(D) & \mathbf{X}(D) \end{array}\right] = \left[\begin{array}{cc|cc} \mathbf{z}_1(D) & \mathbf{z}_1'(D) & \mathbf{x}_1(D) & \mathbf{x}_1'(D) \\ \mathbf{z}_2(D) & \mathbf{z}_2'(D) & \mathbf{x}_2(D) & \mathbf{x}_2'(D) \\ \vdots & \vdots & \vdots & \vdots \\ \mathbf{z}_m(D) & \mathbf{z}_m'(D) & \mathbf{x}_m(D) & \mathbf{x}_m'(D) \end{array}\right]. \tag{72}$$

Every row of the augmented matrix $\mathbf{U}'(D)$ has vanishing symplectic product with itself and any other row. This condition is equivalent to the following matrix condition for shifted symplectic orthogonality [18]:

$$\mathbf{Z}\left(D^{-1}\right)\mathbf{X}(D)^T - \mathbf{X}\left(D^{-1}\right)\mathbf{Z}(D)^T = 0. \tag{73}$$

The construction gives a commuting set of generators for arbitrary shifts and thus forms a valid stabilizer.

We can readily develop a convolutional entanglement distillation protocol using the above formulation. The generators in the augmented matrix $\mathbf{U}'(D)$ correct for errors on the first $n$ ebits. The last $m$ ebits are noiseless ebits that help to obtain a commuting stabilizer. It is necessary to catalyze the distillation protocol with $(n+m)\,\nu$ noiseless ebits. Later frames can use the noiseless ebits generated from previous frames. These initial noiseless ebits are negligible when calculating the yield.

We comment more on the yield of the protocol. The protocol requires a set of $m$ generators with $n+m$ Pauli entries. It generates $n$ ebits for every frame. But it consumes $m$ noiseless ebits per frame. The net yield of a protocol using the above construction is thus $(n-m)/n$.

The key benefit of the above construction is that we can use an arbitrary set of Paulis for distilling noiseless ebits. This arbitrariness in the Paulis implies that we can import an arbitrary classical convolutional binary or quaternary code for use in a convolutional entanglement distillation protocol.

It is again straightforward to develop a noncatastrophic decoding circuit using previous techniques [25]. Every augmented generator in $\mathbf{U}'(D)$ has 1 as an entry so that it satisfies the property required for noncatastrophicity.

*Example 6:* We begin with a classical quaternary convolutional code with entries from $\mathbb{F}_4$:

$$(\cdots|0000|1\bar{\omega}10|1101|0000|\cdots). \tag{74}$$

The above code is a convolutional version of the classical quaternary block code from Ref. [7]. We multiply the above generator by $\bar{\omega}$ and $\omega$ as prescribed in Refs. [5], [20] and use the following map,

| $\mathbb{F}_4$ | $\Pi$ |
|---|---|
| $0$ | $I$ |
| $\omega$ | $X$ |
| $1$ | $Y$ |
| $\bar{\omega}$ | $Z$ |

, (75)

to obtain the following Pauli generators

$$\mathbf{N}(\mathbf{u}_1(D)) = (\cdots|IIII|ZXZI|ZZIZ|IIII|\cdots),$$
$$\mathbf{N}(\mathbf{u}_2(D)) = (\cdots|IIII|XYXI|XXIX|IIII|\cdots). \tag{76}$$

We determine binary polynomials corresponding to the above Pauli generators:

$$\left(\begin{array}{c} \mathbf{u}_1(D) \\ \mathbf{u}_2(D) \end{array}\right) = \left(\begin{array}{cccc|cccc} 1+D & D & 1 & D & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1+D & 1+D & 1 & D \end{array}\right). \tag{77}$$

The first generator anticommutes with itself shifted by one to the left or right, the second generator anticommutes with

itself shifted by one to the left or right, and the first generator anticommutes with the second shifted by one to the left. The following shifted symplectic products confirm the above commutation relations:

$$\begin{aligned}(\mathbf{u}_1 \odot \mathbf{u}_1)(D) &= D^{-1} + D,\\ (\mathbf{u}_2 \odot \mathbf{u}_2)(D) &= D^{-1} + D,\\ (\mathbf{u}_1 \odot \mathbf{u}_2)(D) &= D.\end{aligned} \tag{78}$$

Consider the following two generators:

$$\left(\begin{array}{c}\mathbf{a}_1(D)\\ \mathbf{a}_2(D)\end{array}\right) = \left(\begin{array}{cc|cc} D & 0 & 1 & 0\\ D & D & 0 & 1\end{array}\right). \tag{79}$$

Their relations under the shifted symplectic product are the same as those in (78).

$$\begin{aligned}(\mathbf{a}_1 \odot \mathbf{a}_1)(D) &= (\mathbf{u}_1 \odot \mathbf{u}_1)(D),\\ (\mathbf{a}_2 \odot \mathbf{a}_2)(D) &= (\mathbf{u}_2 \odot \mathbf{u}_2)(D),\\ (\mathbf{a}_1 \odot \mathbf{a}_2)(D) &= (\mathbf{u}_1 \odot \mathbf{u}_2)(D).\end{aligned} \tag{80}$$

We use the construction from (70) so that we have positive delay operators in the augmented matrix. We augment the generators $\mathbf{u}_1(D)$ and $\mathbf{u}_2(D)$ to generators $\mathbf{u}_1'(D)$ and $\mathbf{u}_2'(D)$ respectively as follows. The augmented "Z matrix" is

$$\mathbf{Z}(D) = \left(\begin{array}{cccccc} 1+D & D & 1 & D & D & 0\\ 0 & 1 & 0 & 0 & D & D\end{array}\right), \tag{81}$$

and the augmented "X matrix" is

$$\mathbf{X}(D) = \left(\begin{array}{cccccc} 0 & 1 & 0 & 0 & 1 & 0\\ 1+D & 1+D & 1 & D & 0 & 1\end{array}\right). \tag{82}$$

The augmented matrix $\mathbf{U}'(D)$ is

$$\mathbf{U}'(D) = \left[\begin{array}{c|c} \mathbf{Z}(D) & \mathbf{X}(D)\end{array}\right]. \tag{83}$$

The first row of $\mathbf{U}'(D)$ is generator $\mathbf{u}_1'(D)$ and the second row is $\mathbf{u}_2'(D)$. The augmented generators have the following Pauli representation.

$$\begin{aligned}&\mathbf{N}(\mathbf{u}_1'(D)) =\\ &\qquad(\cdots|IIIIII|ZXZIXI|ZZIZZI|IIIIII|\cdots),\end{aligned}$$

$$\begin{aligned}&\mathbf{N}(\mathbf{u}_2'(D)) =\\ &\qquad(\cdots|IIIIII|XYXIIX|XXIXZZ|IIIIII|\cdots).\end{aligned}$$

The original block code from Ref. [7] corrects for an arbitrary single-qubit error. The above entanglement distillation protocol corrects for a single-qubit error in eight qubits—two frames. This error-correcting capability follows from the capability of the block code. The yield of a protocol using the above stabilizer is again 1/2.

### *D. CSS-Like Construction for a Convolutional Entanglement Distillation Protocol*

We finally present a construction that allows us to import two arbitrary binary classical codes for use in a convolutional entanglement distillation protocol. The construction is similar to a CSS code because one code corrects for bit flips and the other corrects for phase flips.

We could simply use the technique from the previous section to construct a convolutional entanglement-distillation protocol. We could represent both classical codes as codes over $\mathbb{F}_4$. We could multiply the bit-flip code by $\omega$ and the phase-flip code by $\bar{\omega}$ and use the above map from $\mathbb{F}_4$ to the Paulis. We could then use the above method for augmentation and obtain a valid quantum code for entanglement distillation. But there is a better method that exploits the structure of a CSS code to minimize the number of initial catalytic noiseless ebits.

Our algorithm below uses a Gram-Schmidt like orthogonalization procedure to minimize the number of initial noiseless ebits. The procedure is similar to the algorithm in [8] with some key differences.

Suppose we have $m$ generators $\{\mathbf{N}(\mathbf{w}_i(D)) : 1 \leq i \leq m\}$ where

$$\left[\begin{array}{c}\mathbf{w}_1(D)\\ \vdots\\ \mathbf{w}_p(D)\\ \mathbf{w}_{p+1}(D)\\ \vdots\\ \mathbf{w}_m(D)\end{array}\right] = \left[\begin{array}{c|c}\mathbf{z}_1(D) & \mathbf{0}\\ \vdots & \vdots\\ \mathbf{z}_p(D) & \mathbf{0}\\ \mathbf{0} & \mathbf{x}_1(D)\\ \vdots & \vdots\\ \mathbf{0} & \mathbf{x}_{m-p}(D)\end{array}\right]. \tag{84}$$

and each vector $\mathbf{w}_i(D)$ has length $2n$. The above matrix could come from two binary classical codes. The vectors $\mathbf{z}_1(D)$,...,$\mathbf{z}_p(D)$ could come from one code, and the vectors $\mathbf{x}_1(D)$,...,$\mathbf{x}_{m-p}(D)$ could come from another code. The following orthogonality relations hold for the above vectors:

$$\forall\ \ 1 \leq i,j \leq p : (\mathbf{w}_i \odot \mathbf{w}_j)(D) = 0, \tag{85}$$

$$\forall\ \ p+1 \leq i',j' \leq m : (\mathbf{w}_{i'} \odot \mathbf{w}_{j'})(D) = 0. \tag{86}$$

We exploit the above orthogonality relations in the algorithm below.

We can perform a Gram-Schmidt process on the above set of vectors. This process orthogonalizes the vectors with respect to the shifted symplectic product. The procedure does not change the error-correcting properties of the original codes because all operations are linear.

The algorithm breaks the set of vectors above into pairs. Each pair consists of two vectors which are symplectically nonorthogonal to each other, but which are symplectically orthogonal to all other pairs. Any remaining vectors that are symplectically orthogonal to all other vectors are collected into a separate set, which we call the set of isotropic vectors. This idea is similar to the decomposition of a vector space into an isotropic and symplectic part. We cannot label the decomposition as such because the shifted symplectic product is not a true symplectic product.

We detail the initialization of the algorithm. Set parameters $i = 0$, $c = 0$, $l = 0$. The index $i$ labels the total number of vectors processed, $c$ gives the number of pairs, and $l$ labels the number of vectors with no partner. Initialize sets $\mathcal{U}$ and $\mathcal{V}$ to be null: $\mathcal{U} = \mathcal{V} = \emptyset$. $\mathcal{U}$ keeps track of the pairs and $\mathcal{V}$ keeps track of the vectors with no partner.

The algorithm proceeds as follows. While $i \leq m$, let $j \geq 2c + l + 2$ be the smallest index for a $\mathbf{w}_j(D)$ for which $(\mathbf{w}_{2c+l+1} \odot \mathbf{w}_j)(D) \neq 0$. Increment $l$ and $i$ by one, add $i$ to $\mathcal{V}$, and proceed to the next round if no such

pair exists. Otherwise, swap $\mathbf{w}_j(D)$ with $\mathbf{w}_{2c+l+2}(D)$. For $r \in \{2c+l+3, \ldots, m\}$, perform

$$\begin{aligned}\mathbf{w}_r(D) = &(\mathbf{w}_{2c+l+2} \odot \mathbf{w}_{2c+l+1})(D)\,\mathbf{w}_r(D) \\ &+ (\mathbf{w}_r \odot \mathbf{w}_{2c+l+2})\left(D^{-1}\right)\mathbf{w}_{2c+l+1}(D).\end{aligned}$$

if $\mathbf{w}_r(D)$ has a purely $z$ component. Perform

$$\begin{aligned}\mathbf{w}_r(D) = &(\mathbf{w}_{2c+l+1} \odot \mathbf{w}_{2c+l+2})(D)\,\mathbf{w}_r(D) \\ &+ (\mathbf{w}_r \odot \mathbf{w}_{2c+l+1})\left(D^{-1}\right)\mathbf{w}_{2c+l+2}(D).\end{aligned}$$

if $\mathbf{w}_r(D)$ has a purely $x$ component. Divide every element in $\mathbf{w}_r(D)$ by the greatest common factor if the GCF is not equal to one. Then

$$(\mathbf{w}_r \odot \mathbf{w}_{2c+l+1})(D) = (\mathbf{w}_r \odot \mathbf{w}_{2c+l+2})(D) = 0. \quad (87)$$

Increment $c$ by one, increment $i$ by one, add $i$ to $\mathcal{U}$, and increment $i$ by one. Proceed to the next round.

We now give the method for augmenting the above generators so that they form a commuting stabilizer. At the end of the algorithm, the sets $\mathcal{U}$ and $\mathcal{V}$ have the following sizes: $|\mathcal{U}| = c$ and $|\mathcal{V}| = l$. Let us relabel the vectors $\mathbf{w}_i(D)$ for all $1 \leq i \leq 2c+l$. We relabel all pairs: call the first $\mathbf{u}_i(D)$ and call its partner $\mathbf{v}_i(D)$ for all $1 \leq i \leq c$. Call any vector without a partner $\mathbf{u}_{c+i}(D)$ for all $1 \leq i \leq l$. The relabeled vectors have the following shifted symplectic product relations after the Gram-Schmidt procedure:

$$\begin{aligned}&(\mathbf{u}_i \odot \mathbf{v}_j)(D) = f_i(D)\,\delta_{ij} \quad \forall \;\; i,j \in \{1,\ldots,c\}, \\ &(\mathbf{u}_i \odot \mathbf{u}_j)(D) = 0 \qquad\quad \forall \;\; i,j \in \{1,\ldots,l\}, \\ &(\mathbf{v}_i \odot \mathbf{v}_j)(D) = 0 \qquad\quad \forall \;\; i,j \in \{1,\ldots,c\}, \end{aligned} \quad (88)$$

where $f_i(D)$ is an arbitrary polynomial. Let us arrange the above generators in a matrix as follows:

$$\begin{bmatrix} \mathbf{u}_1(D) \\ \vdots \\ \mathbf{u}_c(D) \\ \mathbf{v}_1(D) \\ \vdots \\ \mathbf{v}_c(D) \\ \mathbf{u}_{c+1}(D) \\ \vdots \\ \mathbf{u}_{c+l}(D) \end{bmatrix}. \quad (89)$$

We augment the above generators with the following matrix so that all vectors are orthogonal to each other:

$$\left[\begin{array}{cccc|c} f_1\left(D^{-1}\right) & 0 & \cdots & 0 & \mathbf{0}_{1\times c} \\ 0 & f_2\left(D^{-1}\right) & & \vdots & \mathbf{0}_{1\times c} \\ \vdots & & \ddots & 0 & \vdots \\ 0 & \cdots & 0 & f_c\left(D^{-1}\right) & \mathbf{0}_{1\times c} \\ \mathbf{0}_{c\times 1} & \mathbf{0}_{c\times 1} & \cdots & \mathbf{0}_{c\times 1} & \mathbf{I}_{c\times c} \\ \mathbf{0}_{l\times 1} & \mathbf{0}_{l\times 1} & \cdots & \mathbf{0}_{l\times 1} & \mathbf{0}_{l\times c} \end{array}\right]. \quad (90)$$

The yield of a protocol using the above construction is $(n-m)/n$. Suppose we use an $[n, k_1]$ classical binary convolutional code for the bit flips and an $[n, k_2]$ classical binary convolutional code for the phase flips. Then the convolutional entanglement distillation protocol has yield $(k_1 + k_2 - n)/n$.

*Example 7:* Consider a binary classical convolutional code with the following parity check matrix:

$$\begin{bmatrix} 1+D & D & 1 \end{bmatrix}. \quad (91)$$

We can use the above parity check matrix to correct both bit and phase flip errors in an entanglement distillation protocol. Our initial quantum parity check matrix is

$$\left[\begin{array}{ccc|ccc} 1+D & D & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1+D & D & 1 \end{array}\right]. \quad (92)$$

The shifted symplectic product for the first and second row is $D^{-1} + D$. We therefore augment the above matrix as follows:

$$\left[\begin{array}{cccc|cccc} 1+D & D & 1 & D^{-1}+D & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1+D & D & 1 & 1 \end{array}\right]. \quad (93)$$

The above matrix gives a valid stabilizer for use in an entanglement distillation protocol. The yield of a protocol using the above stabilizer is 1/3.

## V. Conclusion and Current Work

We constructed a theory of convolutional entanglement distillation. The entanglement-assisted protocol assumes that the sender and receiver have some noiseless ebits to use as a catalyst for distilling more ebits. These protocols have the benefit of lifting the self-orthogonality constraint. Thus we are able to import an arbitrary classical convolutional code for use in a convolutional entanglement distillation protocol. The error-correcting properties and rate of the classical code translate to the quantum case. Brun, Devetak, and Hsieh first constructed the method for importing an arbitrary classical block code in their work on entanglement-assisted codes [8], [7]. Our theory of convolutional entanglement distillation paves the way for exploring protocols that approach the optimal distillable entanglement by using the well-established theory of classical convolutional coding.

Convolutional entanglement distillation protocols also hold some key advantages over block entanglement distillation protocols. They have a higher yield of ebits, lower decoding complexity, and are an online protocol that a sender and receiver can employ as they acquire more noisy ebits.

We suggest that convolutional entanglement distillation protocols may bear some advantages for distillation of a secret key because of the strong connection between distillation and privacy [14]. We are currently investigating whether convolutional entanglement distillation protocols can improve the secret key rate for quantum key distribution.

The authors thank Igor Devetak and Zhicheng Luo for useful discussions and thank Saikat Guha for locating a copy of Jonsson's master's thesis. MMW acknowledges support from NSF Grant 0545845, and HK and TAB acknowledge support from NSF Grant CCF-0448658. All authors are grateful to the Hearne Insitute for Theoretical Physics for hosting MMW as a visiting researcher.

## VI. Appendix

*Example 8:* We present an example of an entanglement-assisted code that corrects an arbitrary single-qubit error [7]. Suppose the sender wants to use the quantum error-correcting properties of the following nonabelian subgroup of $\Pi^4$:

$$\begin{array}{cccc} Z & X & Z & I \\ Z & Z & I & Z \\ X & Y & X & I \\ X & X & I & X \end{array} \tag{94}$$

The first two generators anticommute. We obtain a modified third generator by multiplying the third generator by the second. We then multiply the last generator by the first, second, and modified third generators. The error-correcting properties of the generators are invariant under these operations. The modified generators are as follows:

$$\begin{array}{rcccc} g_1 & = & Z & X & Z & I \\ g_2 & = & Z & Z & I & Z \\ g_3 & = & Y & X & X & Z \\ g_4 & = & Z & Y & Y & X \end{array} \tag{95}$$

The above set of generators have the commutation relations given by the fundamental theorem of symplectic geometry:

$$\begin{aligned} &\{g_1, g_2\} = [g_1, g_3] = [g_1, g_4], \\ &= [g_2, g_3] = [g_2, g_4] = [g_3, g_4] = 0. \end{aligned}$$

The above set of generators is unitarily equivalent to the following canonical generators:

$$\begin{array}{cccc} X & I & I & I \\ Z & I & I & I \\ I & Z & I & I \\ I & I & Z & I \end{array} \tag{96}$$

We can add one ebit to resolve the anticommutativity of the first two generators:

$$\left[\begin{array}{cccc|c} X & I & I & I & X \\ Z & I & I & I & Z \\ I & Z & I & I & I \\ I & I & Z & I & I \end{array}\right] \tag{97}$$

The following state is an eigenstate of the above stabilizer

$$\left|\Phi^+\right\rangle^{AB} |00\rangle^A |\psi\rangle^A. \tag{98}$$

where $|\psi\rangle^A$ is a qubit that the sender wants to encode. The encoding unitary then rotates the generators in (97) to the following set of globally commuting generators:

$$\left[\begin{array}{cccc|c} Z & X & Z & I & X \\ Z & Z & I & Z & Z \\ Y & X & X & Z & I \\ Z & Y & Y & X & I \end{array}\right] \tag{99}$$

The receiver measures the above generators upon receipt of all qubits to detect and correct errors.

### A. Encoding Algorithm

We continue with the previous example. We detail an algorithm for determining an encoding circuit and the optimal number of ebits for the entanglement-assisted code. The operators in (94) have the following representation as a binary matrix:

$$H = \left[\begin{array}{cccc|cccc} 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{array}\right]. \tag{100}$$

Call the matrix to the left of the vertical bar the "$Z$ matrix" and the matrix to the right of the vertical bar the "$X$ matrix."

The algorithm consists of row and column operations on the above matrix. Row operations do not affect the error-correcting properties of the code but are crucial for arriving at the optimal decomposition from the fundamental theorem of symplectic geometry. The operations available for manipulating columns of the above matrix are Clifford operations [4]. Clifford operations preserve the Pauli group $\Pi^n$ under conjugation. The CNOT gate, the Hadamard gate, and the Phase gate generate the Clifford group. A CNOT gate from qubit $i$ to qubit $j$ adds column $i$ to column $j$ in the $X$ matrix and adds column $j$ to column $i$ in the $Z$ matrix. A Hadamard gate on qubit $i$ swaps column $i$ in the $Z$ matrix with column $i$ in the $X$ matrix and vice versa. A phase gate on qubit $i$ adds column $i$ in the $X$ matrix to column $i$ in the $Z$ matrix. Three CNOT gates implement a qubit swap operation [15]. The effect of a swap on qubits $i$ and $j$ is to swap columns $i$ and $j$ in both the $X$ and $Z$ matrix.

The algorithm begins by computing the symplectic product between the first row and all other rows. We emphasize that the symplectic product here is the standard symplectic product. Leave the matrix as it is if the first row is not symplectically orthogonal to the second row or if the first row is symplectically orthogonal to all other rows. Otherwise, swap the second row with the first available row that is not symplectically orthogonal to the first row. In our example, the first row is not symplectically orthogonal to the second so we leave all rows as they are.

Arrange the first row so that the top left entry in the $X$ matrix is one. A CNOT, swap, Hadamard, or combinations of these operations can achieve this result. We can have this result in our example by swapping qubits one and two. The matrix becomes

$$\left[\begin{array}{cccc|cccc} 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{array}\right]. \tag{101}$$

Perform CNOTs to clear the entries in the $X$ matrix in the top row to the right of the leftmost entry. These entries are already zero in this example so we need not do anything. Proceed to the clear the entries in the first row of the $Z$ matrix. Perform a phase gate to clear the leftmost entry in the first row of the $Z$ matrix if it is equal to one. It is equal to zero in this case so we need not do anything. We then use Hadamards

and CNOTs to clear the other entries in the first row of the $Z$ matrix.

We perform the above operations for our example. Perform a Hadamard on qubits two and three. The matrix becomes

$$\left[\begin{array}{cccc|cccc} 0&0&0&0&1&1&1&0\\ 1&0&0&1&0&1&0&0\\ 1&1&1&0&1&0&0&0\\ 0&1&0&0&1&0&0&1 \end{array}\right]. \tag{102}$$

Perform a CNOT from qubit one to qubit two and from qubit one to qubit three. The matrix becomes

$$\left[\begin{array}{cccc|cccc} 0&0&0&0&1&0&0&0\\ 1&0&0&1&0&1&0&0\\ 1&1&1&0&1&1&1&0\\ 1&1&0&0&1&1&1&1 \end{array}\right]. \tag{103}$$

The first row is complete. We now proceed to clear the entries in the second row. Perform a Hadamard on qubits one and four. The matrix becomes

$$\left[\begin{array}{cccc|cccc} 1&0&0&0&0&0&0&0\\ 0&0&0&0&1&1&0&1\\ 1&1&1&0&1&1&1&0\\ 1&1&0&1&1&1&1&0 \end{array}\right]. \tag{104}$$

Perform a CNOT from qubit one to qubit two and from qubit one to qubit four. The matrix becomes

$$\left[\begin{array}{cccc|cccc} 1&0&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0\\ 0&1&1&0&1&0&1&1\\ 1&1&0&1&1&0&1&1 \end{array}\right]. \tag{105}$$

The first two rows are now complete. They need one ebit to compensate for their anticommutativity or their nonorthogonality with respect to the symplectic product.

Now we perform a "Gram-Schmidt orthogonalization" with respect to the symplectic product. Add row 1 to any other row that has one as the leftmost entry in its $Z$ matrix. Add row two to any other row that has one as the leftmost entry in its $X$ matrix. For our example, we add row one to row four and we add row two to rows three and four. The matrix becomes

$$\left[\begin{array}{cccc|cccc} 1&0&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0\\ 0&1&1&0&0&0&1&1\\ 0&1&0&1&0&0&1&1 \end{array}\right]. \tag{106}$$

The first two rows are now symplectically orthogonal to all other rows per the fundamental theorem of symplectic geometry.

We proceed with the same algorithm on the next two rows. The next two rows are symplectically orthogonal to each other so we can deal with them individually. Perform a Hadamard on qubit two. The matrix becomes

$$\left[\begin{array}{cccc|cccc} 1&0&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0\\ 0&0&1&0&0&1&1&1\\ 0&0&0&1&0&1&1&1 \end{array}\right]. \tag{107}$$

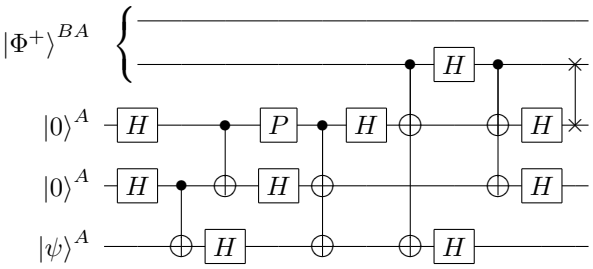


Fig. 7. Encoding circuit for the entanglement-assisted code from [7]. The "H" gate is a Hadamard gate and the "P" gate is a phase gate.

Perform a CNOT from qubit two to qubit three and from qubit two to qubit four. The matrix becomes

$$\left[\begin{array}{cccc|cccc} 1&0&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0\\ 0&1&1&0&0&1&0&0\\ 0&1&0&1&0&1&0&0 \end{array}\right]. \tag{108}$$

Perform a phase gate on qubit two:

$$\left[\begin{array}{cccc|cccc} 1&0&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0\\ 0&0&1&0&0&1&0&0\\ 0&0&0&1&0&1&0&0 \end{array}\right]. \tag{109}$$

Perform a Hadamard on qubit three followed by a CNOT from qubit two to qubit three:

$$\left[\begin{array}{cccc|cccc} 1&0&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0\\ 0&0&0&0&0&1&0&0\\ 0&0&0&1&0&1&1&0 \end{array}\right]. \tag{110}$$

Add row three to row four and perform a Hadamard on qubit two:

$$\left[\begin{array}{cccc|cccc} 1&0&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0\\ 0&1&0&0&0&0&0&0\\ 0&0&0&1&0&0&1&0 \end{array}\right]. \tag{111}$$

Perform a Hadamard on qubit four followed by a CNOT from qubit three to qubit four. End by performing a Hadamard on qubit three:

$$\left[\begin{array}{cccc|cccc} 1&0&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0\\ 0&1&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0 \end{array}\right]. \tag{112}$$

The above matrix now corresponds to the canonical Paulis (96). Adding one half of an ebit to the receiver's side gives the canonical stabilizer (97) whose simultaneous +1-eigenstate is (98).

Figure 7 gives the encoding circuit corresponding to the above operations. The above operations in reverse order take the canonical stabilizer (97) to the encoded stabilizer (99).

## REFERENCES


[1] P. W. Shor, "Scheme for reducing decoherence in quantum computer memory," *Phys. Rev. A*, vol. 52, no. 4, pp. R2493–R2496, Oct 1995.

[2] A. R. Calderbank and P. W. Shor, "Good quantum error-correcting codes exist," *Phys. Rev. A*, vol. 54, no. 2, pp. 1098–1105, Aug 1996.

[3] A. M. Steane, "Error correcting codes in quantum theory," *Phys. Rev. Lett.*, vol. 77, no. 5, pp. 793–797, Jul 1996.

[4] D. Gottesman, "Stabilizer codes and quantum error correction," Ph.D. dissertation, California Institue of Technology, 1997.

[5] A. Calderbank, E. Rains, P. Shor, and N. Sloane, "Quantum error correction via codes over gf(4)," *IEEE Trans. Inf. Theory*, vol. 44, pp. 1369–1387, 1998.

[6] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*. North Holland, 1983.

[7] T. A. Brun, I. Devetak, and M.-H. Hsieh, "Correcting quantum errors with entanglement," *Science*, vol. 314, no. 5798, pp. pp. 436 – 439, October 2006.

[8] ——, "Catalytic quantum error correction," *arXiv:quant-ph/0608027*, August 2006.

[9] D. Gottesman. [Online]. Available: http://www.perimeterinstitute.ca/personal/dgottesman/QECC2007/Sols9.pdf

[10] C. H. Bennett, G. Brassard, S. Popescu, B. Schumacher, J. A. Smolin, and W. K. Wootters, "Purification of noisy entanglement and faithful teleportation via noisy channels," *Phys. Rev. Lett.*, vol. 76, no. 5, pp. 722–725, Jan 1996.

[11] C. H. Bennett, D. P. DiVincenzo, J. A. Smolin, and W. K. Wootters, "Mixed-state entanglement and quantum error correction," *Phys. Rev. A*, vol. 54, no. 5, pp. 3824–3851, Nov 1996.

[12] C. H. Bennett and S. J. Wiesner, "Communication via one- and two-particle operators on einstein-podolsky-rosen states," *Phys. Rev. Lett.*, vol. 69, no. 20, pp. 2881–2884, Nov 1992.

[13] C. H. Bennett, G. Brassard, C. Crépeau, R. Jozsa, A. Peres, and W. K. Wootters, "Teleporting an unknown quantum state via dual classical and einstein-podolsky-rosen channels," *Phys. Rev. Lett.*, vol. 70, no. 13, pp. 1895–1899, Mar 1993.

[14] P. W. Shor and J. Preskill, "Simple proof of security of the bb84 quantum key distribution protocol," *Phys. Rev. Lett.*, vol. 85, no. 2, pp. 441–444, Jul 2000.

[15] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*. Cambridge University Press, 2000.

[16] Z. Luo and I. Devetak, "Efficiently implementable codes for quantum key expansion," *Phys. Rev. A*, vol. 75, no. 1, p. 010303, 2007.

[17] H. Ollivier and J.-P. Tillich, "Description of a quantum convolutional code," *Phys. Rev. Lett.*, vol. 91, no. 17, p. 177902, Oct 2003.

[18] ——, "Quantum convolutional codes: fundamentals," *arXiv:quant-ph/0401134*, 2004.

[19] J. G. David Forney and S. Guha, "Simple rate-1/3 convolutional and tail-biting quantum error-correcting codes," in *IEEE International Symposium on Information Theory (arXiv:quant-ph/0501099)*, 2005.

[20] G. D. Forney, M. Grassl, and S. Guha, "Convolutional and tail-biting quantum error-correcting codes," *IEEE Trans. Inf. Theory*, vol. 53, pp. 865–880, 2007.

[21] R. Johannesson and K. S. Zigangirov, *Fundamentals of Convolutional Coding*. Wiley-IEEE Press, 1999.

[22] H. F. Chau, "Quantum convolutional error-correcting codes," *Phys. Rev. A*, vol. 58, no. 2, pp. 905–909, Aug 1998.

[23] ——, "Good quantum-convolutional error-correction codes and their decoding algorithm exist," *Phys. Rev. A*, vol. 60, no. 3, pp. 1966–1974, Sep 1999.

[24] M. Grassl and M. Rötteler, "Quantum convolutional codes: Encoders and structural properties," in *Forty-Fourth Annual Allerton Conference*, 2006.

[25] ——, "Noncatastrophic encoders and encoder inverses for quantum convolutional codes," in *IEEE International Symposium on Information Theory (quant-ph/0602129)*, 2006.

[26] A. C. da Silva, *Lectures on Symplectic Geometry*. Springer, 2001.

[27] I. Devetak, A. W. Harrow, and A. Winter, "A resource framework for quantum shannon theory," *arXiv:quant-ph/0512015*, 2005.

[28] A. J. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Trans. Inf. Theory*, vol. 13, pp. 260–269, 1967.